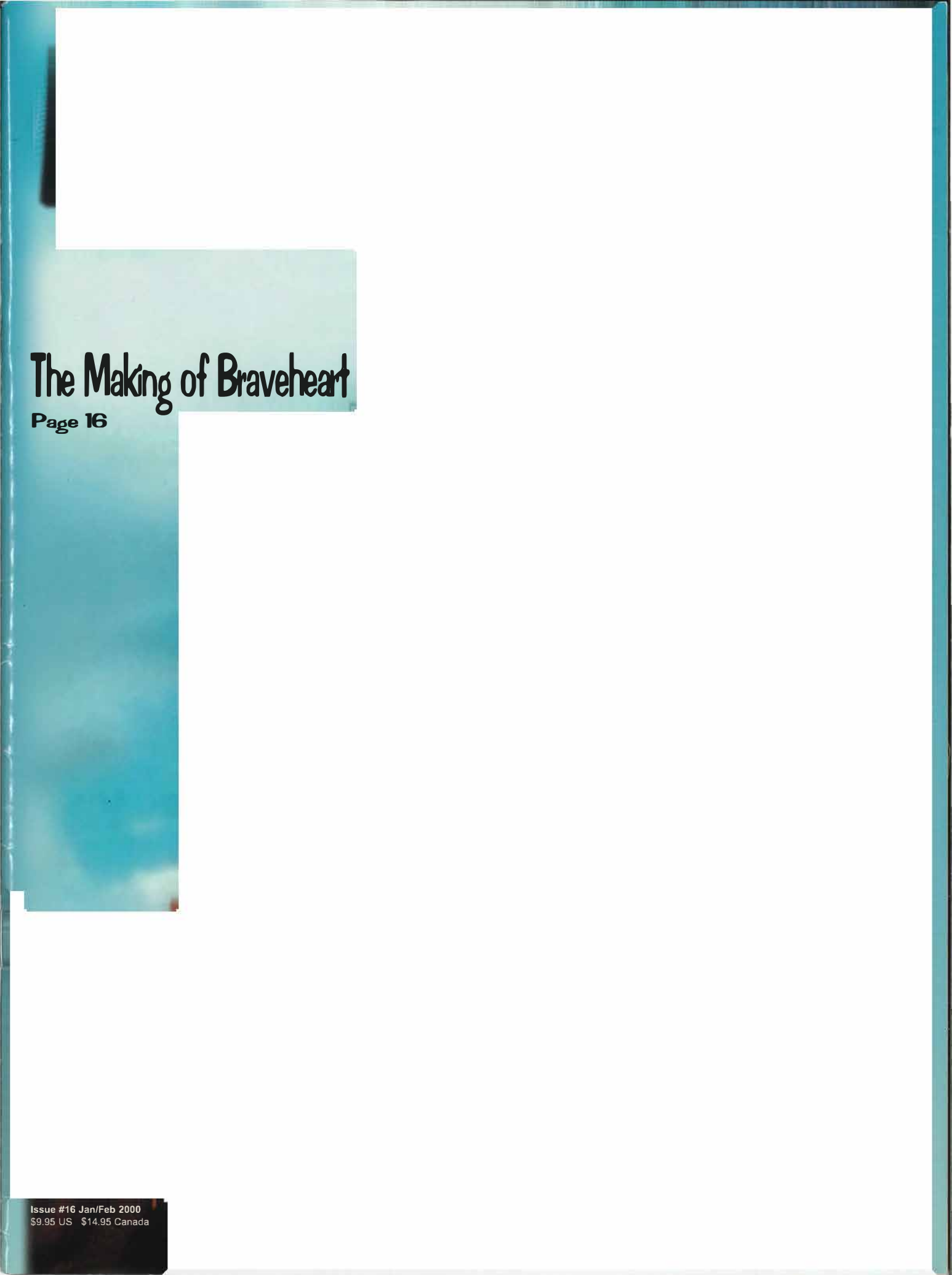


The Making of Braveheart

Page 16

Issue #16 Jan/Feb 2000
$9.95 US $14.95 Canada

LightWave® [6]

ARETE'S

NATUREFX FOR LIGHTWAVE 3D

ATMOSPHERE | CLOUDS | WATER | WAKES

Allow yourself to be seduced.

Seduced by the power, the elegance and the flexibility. Your first encounter will reveal why LightWave dominates the demanding world of show business.

LightWave has always been the industry standard for exceptional realism. Today it just got better. Much better. Now, with caustics, radiosity and integrated Hypervoxels, LightWave [6] understands and delivers the nuances of lighting and volumetrics better than any other animation system on the market. The addition of a 96-bit floating-point render pipeline assures you ultimate color depth and accuracy.

LightWave [6] introduces the world's fastest and most accurate hybrid inverse/forward kinematics engine. Also, unique to LightWave [6] are IntelligEntities™. These include Endomorphs™ and Skelegons™ used to create and control character animation with efficiency and fluidity.

With the release of LightWave [6] nothing was left to chance. We completely redesigned the program and the process with you, the artist, in mind. The joy of LightWave, all the features, all the functions — better, faster, fun.

The seduction is complete.

NewTek booth #535 at Siggraph '99

www.newtek.com
Call **(800) 862-7837**
Outside the U.S. call (210) 370-8000

LightWave 3D is a registered trademark of NewTek, Inc. Hypervoxels, IntelligEntities, Endomorphs and Skelegons are trademarks of NewTek, Inc. © NewTek, 1999. All rights reserved.

www.sharbor.com

800-944-5520 • 702-990-2277
WWW.DYNAMIC-REALITIES.COM

KEYFRAME MAGAZINE

The Editor
Doug Kelly *dakelly@megalink.net*

Advertising Rep
Tiffani Banaszak *keyframe1@aol.com*

Art Director
Tiffani Banaszak *keyframe1@aol.com*

Co-Conspirators, Writers, and Artists
Jason Goodman *jason@msd-nyc.com*
Brenden Mecleary *bman2@ix.netcom.com*
Laurie McCulloch
laurie@digital-animations.com
Matt Morgan *matt@threshold-digital.com*
Steve Pugh *steve@foundation-i.com*
John Riggs *jriggs@vision-scape.com*
Lee Stranahan *lee@stranahan.com*
Phil South *phil_south@hotmail.com*
David Warner *dwarner@albany.net*

The Digital Underground
DMG Publishing
2756 N. Green Valley Pkwy. #261
Henderson, NV 89014
702-990-8656

How To Get It
888-778-WAVE
http://www.keyframemag.com

Keyframe Magazine is published 6 times annually by DMG Publishing. © 2000 DMG Publishing. All rights reserved. Reproduction, in part or in whole, without written permission is strictly prohibited. Printed in the USA.

The Secret Formula
Keyframe Magazine is created on Intergraph machines using Quark 4.1, Photoshop 4.0, LightWave 5.6, Painter 5, and various plug-ins. Hardware includes a MusTek 2400 9600dpi scanner, Iomega Zip and Jaz drives, and a DPS Perception.

No dongles were harmed in the making of this publication.

Departments

3 Editorial by Doug Kelly
- Waiting for the other shoe to drop

4 Ear to the Wall
- Wild Brain's Animation Showcase
- DNA Productions to Produce *Jimmy Neutron, Boy Genius*
- Joe Alter Wins Animation Award
- Shave and a Haircut Now Available

12 On Topic by David Warner
- L6, Speed, Illustrator, Thicken, CMYK, Memory

14 Macintosh Up Close & Personal by Jason Goodman
- Stereoscopic Video

26 Take One by Lee Stranahan
- LightWave Production Tips

31 A Look at Deep Paint by Doug Kelly

32 Final Cut by Steve Pugh
- Bytes Behind the Roughnecks

Tutorials

6 Character Shop by Doug Kelly
- Adapting Models

9 Matching Real World Lighting: Part 2 by Matt Morgan
- Matching Plate Photography

16 The Making of Braveheart by Laurie McCulloch
- Putting Your Best Face Forward

20 Bone & Expressions in Project: messiah by John Riggs
- Animating a Hand

24 Compositing with the Enemy by Phil South
- Compositing LightWave Objects

28 Programmer's Hideaway by Brenden MeCleary
- LightWave Envelope Support for Plug-ins

EDITORIAL

Waiting for the Other Shoe to Drop

Patience is a virtue, and persistence is the key to success. Yeah, right! It's hard to take a philosophical view of events when life isn't going the way you want it to. Personally, I feel very good about the potential for the entire NewTek line and related third-party market for the next year, but I can't say I'm pleased with the delivery schedule. I want my toys/tools now, not next month! I'm putting my short film back into production after a yearlong hiatus, and I'm really looking forward to using the new software to wrap up the little monster.

2000 is looking like a very good year for the NewTek family. VTNT, L6, and Aura are all shipping major new versions or at least significant upgrades in the near future. Aura 2.0 sounds like a major improvement, with rumors of support for tracking and better chroma keying. Those two capabilities alone will really pull Aura into the mainstream of digital compositing software, and will reinforce the viability of NewTek's VTNT/LightWave/Aura/Razor as an integrated video production solution. The impending release of the Switcher won't hurt either. I'm very much looking forward to news from NAB in April.

The L6 open public beta is a good thing. A few soreheads have complained, but based on my (blessedly brief) experience in 3D software development, I believe NewTek has made the best choice available. The more people who get to bang on the software as long as possible, the more robust the shipping product will be. NewTek has been very up-front about the stability and speed issues of the beta release, and the company is commendably insistent that beta testers *not* use it in production. I've been beating the heck out of my copy, and I reported the surprisingly small number of bugs I managed to locate. So far, I like what I see. Once it's stable and optimized, I think L6 will be able to stand up to any software in its class.

While we're all waiting for the other shoe to drop, this issue presents plenty of new and useful information to keep everyone busy. The cover case study goes under the kilt - oops, I mean behind the scenes - of DigiMania's work on the Braveheart game, showing how they created a CGI Mel Gibson. Our first tutorial using Station X Studios' Project: Messiah plug-in is contributed by John Riggs, who has written an excellent piece on setting up bones and expressions for flexible and efficient animation of a hand. Matt Morgan wraps up his comprehensive two-part tutorial on matching background plate lighting in LightWave; and on a related topic, Phil South shows how to composite UFOs into a variety of real-world lighting environments. For you technical directors, Brenden Mecleary explains how to use the LightWave SDK's Envelopes plug-in to animate your own plug-in's parameters. In Character Shop, I demonstrate how to recycle an overly complex model into a more efficient polygonal or NURBS model. David Warner has selected his usual collection of useful gems from the LightWave Mailing List, and Lee Stranahan presents some handy tips for keeping your productions organized. Jason Goodman shows off a new gadget for shooting stereoscopic video, and he discusses some possibilities for creating stereo 3D effects in LightWave. Another new tool reviewed in this issue is Deep Paint 3D, a handy means of texturing your models. Last but not least, Steve Pugh gets philosophical about the wrapping of *Starship Troopers*.

I hope this issue presents enough interesting possibilities to keep you busy until the new tools arrive.

Doug Kelly

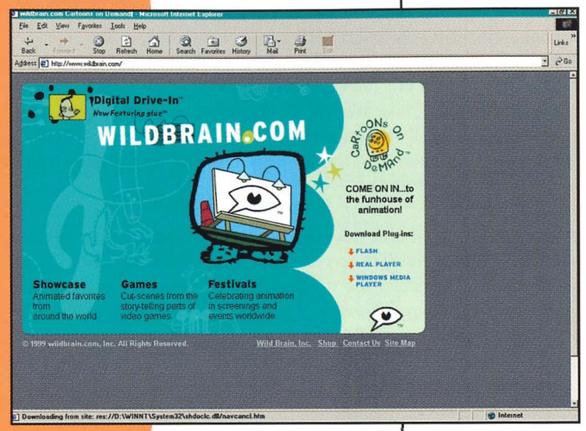
Ear to the Wall

News from the LightWave arena

Wildly on My Mind...

Wild Brain, Inc., has announced the launch or wildbrain.com, their new Web channel showcasing the best in animation programming from around the world. wildbrain.com features on-demand video streaming available through a broadcast services agreement with Yahoo! Inc..

Wild Brain, Inc.'s talent base of Oscar, Emmy and Clio winning artists are renowned for their innovative convergence of art and technology in film, television series, commercials and new media productions. The studio will launch wildbrain.com with their original animated series, *Glue*, an internet exclusive that features colorful characters in their quirky 2D town.



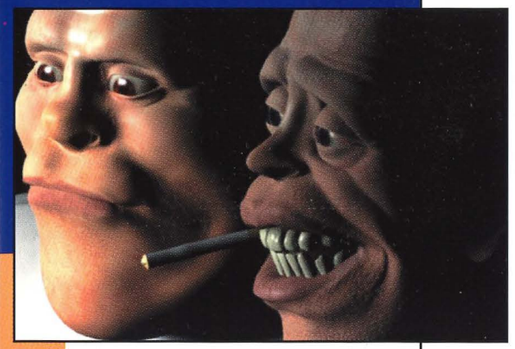
wildbrain.com is a visual feast of some of the world's best animations featuring four key areas: Digital Drive-In, where original animation produced exclusively for wildbrain.com is shown, Festivals, with content, interviews and live broadcasts from around the globe, Showcase, which has animated projects from every corner of the universe, and Games, featuring "cut scenes" previously available only to expert players.

wildbrain.com is free to Web users and promises to be the definitive source for animation on the Internet. Dedicated solely to the global promotion of the art of animation, their unique new channel will broadcast ANIMATION ONLY - in all of its forms - something not available anywhere else on the Net.

DNA to Produce *Jimmy Neutron, Boy Genius*

DNA Productions, Inc., has entered into a multi-year agreement with Nickelodeon and screenwriter and director Steve Oedekerk to produce *Jimmy Neutron, Boy Genius*, Nickelodeon's first property to be simultaneously developed and produced as a feature film, television series and online franchise.

Originally created by John Davis of DNA Productions, *Jimmy Neutro, Boy Genius* will be Nickelodeon's first 3D, CGI animated character, and its first property to be developed as a multi-media franchise. The film, television series and online component - which includes online games for www.nick.com - will be produced by Nickelodeon Movies, Nickelodeon Productions, Oedekerk's O Entertainment, and DNA Productions, Inc.. The feature film will be distributed by Paramount Pictures. Nickelodeon will debut Jimmy Neutron as a series of one-minute cartoon shorts on the network prior to theatrical release, then, the show will join Nickelodeon's line-up as a series.



"Jersey" Wins at IMAGINA

Feb 2, Monaco - Up against industry giants PIXAR/DISNEY for "Toy Story II", and former employer Industrial Light and Magic for "Episode 1 : The Phantom Mance", animator Joe Alter and his company Joe Alter, Inc. won at Imagina for "Best Synthetic Actor".

Funded on a shoestring by Film Roman, Inc. (known for "The Simpsons"), Alter and Film Roman Creative Director Eric Redomski, collaborated for 5 months to create the experimental "Jersey", an animated short about two garbage men.

'It's sort of "The Sopranos" meets "The Honeymooners," says Alter who directs and animates. "You get two Jersey working guys in a truck and see what happens. I really wanted to demonstrate the capability of my new facial animation software and have two characters just sit there and relate to each other in simple human terms. We really haven't seen that before in animation; usually there's all sorts of crazy action to cover up the fact that it's so hard to animate really expressive faces. I really could not be more thrilled by this award."

Imagina, a computer graphics film festival held every February in Monaco Fr, takes entries from around the world and tabulates a peer vote.

Alter is known primarily for his 14 years of visual effects work on productions such as "The Mask", "Cliffhanger", "Godzilla", "The Prince of Egypt", and "Star Trek : TNG". He has also contributed to two sci-tech Oscars, and has worked for Industrial Light and Magic, Dreamworks, and Warner Bros.

Redomski is known for his work as co-creator of the Batman cartoon series, and has produced for Ralph Bakshi, and directed on SpawnII.

For "Jersey", a 3.5 minute animated short produced for 35mm on a single pentium III with Lightwave and LipService, to win such a prestigious award is a shocking upset. "I really never expected to compete in this category." Says Alter,"For it to win is almost unfathomable."

For more information and to view "Jersey", visit www.joealter.com.

Shave and A Haircut!

Bringing high end hair rendering, modelling, and physical dynamics to LightWave, this modestly priced tool looks like a winner. Shave and a Haircut, by Joe Alter Inc., comes with everything you'll need to create photorealistic hair with LightWave.

The tools in Shave's standalone package allow you to create virtually any hair style in minutes. Displacement plug-ins in Layout allow you to stick the hair to your moving/flexing object in Layout, and a pixel filter plug-in contains the render. Simple intuitive adjustments allow the user to create frizzy, curly, and wavy hair. The dynamics also maintain the hair's character (kink, clump) as they move, so it doesn't become a loose mop of spaghetti when the dynamics are applied.

Shave employs a breakthrough new rendering algorithm which requires very little memory, as low as 32MB. It self-shadows on a hair-by-hair level (most others fake this shadowing with an offset of the geometry or coloring roots dark), it colors strands with proceedural variation per strand (creating rich colors), and does transparency. Backlighting a model creates exactly what you would expect, a halo of lit hair,

The last year has seen a dramatic growth in the number of games, television and film projects using humanoid characters at near- or photorealistic levels of modeling and texturing. The demand for this level of work will be consistent in the future, because once the public (and clients) knows it can be done, they will continue to require it. That means you'd better learn to model and texture photorealistic human characters. If you already know how, you'd better learn more ways to make the process as streamlined and efficient as possible.

One way to boost your productivity is by recycling models. I don't mean using the same generic head and body for every character, and only modifying the textures. Although that approach is still viable for very limited game work, it won't pass muster with higher-end game, TV or film audiences. Instead, you can employ several techniques to transform a stock character into a markedly different, apparently unique one. Used properly, these techniques can be much more time- and cost-effective than the process of creating a new character from scratch.

The fastest, most efficient method for a talented and experienced LightWave modeler to create a unique character for personal use is to model it directly in the computer, either from character design drawings or direct inspiration. However, if the character is to be used by a production team, approved by a client, director or other decision-maker, is one of several characters in a production, or the modeler is not talented and/or experienced enough, a different approach to modeling will be more efficient.

In *Keyframe #9* I discussed the benefits and detailed the process of sculpting a maquette. To briefly review: a physical maquette is more efficient to create for traditionally-trained sculptors; clients, directors and other decision-makers intuitively understand maquettes far better than CGI models; and a maquette is a physical deliverable that the client can sign off and the rest of the production team can refer to, thereby preventing or minimizing many arguments, changes and other problems. Maquettes are a good thing in almost any production, and I recommend them for personal solo projects as well.

The manual photogrammetry process I described in *Keyframe #9* is a reliable and cost-effective one, and will produce a reasonably accurate low-polygon mesh that is efficient to set up and animate. However, this process does take more time than you may have in a commercial production with client-imposed deadlines. This is an example of the truism that "When you have more money, you have less time." The solution, appropriately, is to throw a little money at the problem. Build a maquette, and send it out to a service bureau that uses one of Cyberware's laser scanners. Because you only need the raw surface data, you can negotiate for the service bureau's minimum setup and scan charges. You don't need to pay them a fortune for some operator's idea of an efficient NURBS surface.

What you'll get back is a very heavy and very accurate polygonal model like that shown in Figure 1. This is a Cyberscan of my Fred maquette, scanned and cleaned up by Sue Addleman at Cyberware. Sue is probably one of the most experienced and capable Cyberscanner operators in the world, and did a very thorough job of integrating several partial scans and closing any gaps. You would pay

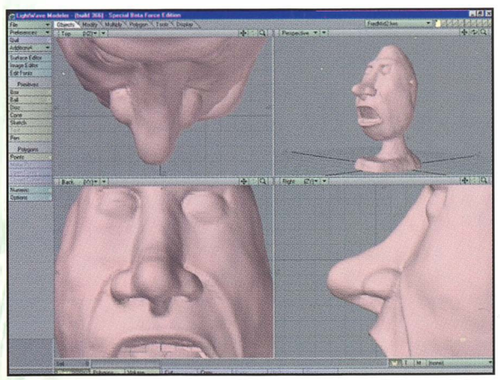


Figure 1: Fred maquette, Cyberscanned and loaded into Modeler. Note that the model's surface in the smooth shaded perspective view is almost indistinguishable from that in the flat shaded orthographic views, due to the very high polygon count.

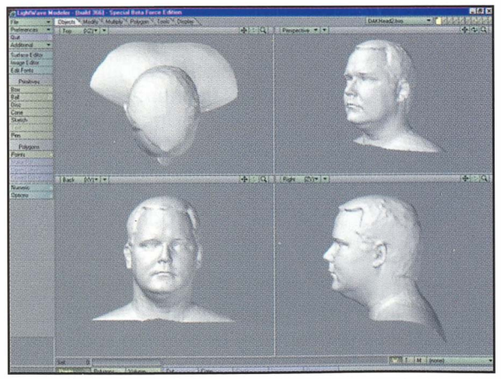


Figure 2: My head, Cyberscanned and loaded into Modeler. Scanning by David Addleman, CyberWare.

higher for a scan of this quality, but since the rest of this process is tolerant of gaps and partial scans, you can still work with the cheaper raw scan data.

If you will be working only with rigid maquettes, you may also consider the purchase or occasional subcontracted use of a manual digitizer such as the Microscribe. The disadvantages of this tool are that it takes a little longer, is more prone to errors of omission that aren't as obvious as a hole in a Cyberscan, and you can't digitize live subjects. If your work will include duplicating human models like the Mel Gibson character for the Braveheart game in this issue's cover case study, you'll need to use a Cyberscan. A scan of a human actor's head will look something like Figure 2, and will include a matching color map.

The increased efficiency of adapting an existing model over creating one from scratch is especially true for humanoid heads, which usually have similar features and require only some repositioning of existing points rather than a completely new mesh. Once you have created a head model that is useful for emotional expressions and lip sync, there is no reason you should not recycle that model in order to create other characters. This approach can also enable you to recycle morph libraries and other animation tools from one character to another, changing only what is necessary rather than starting from scratch. You can apply these same techniques to reshape an existing model over a Cyberscan form of a maquette or a live actor.

One of the tools you can use for the reshaping process is Dave Hudek's Impressions plug-in, which I reviewed in Keyframe #9. This Modeler plug-in is something like shrink-wrap, but controllable. You load the form model as a background object, load the model to be recycled in the foreground, then use Impressions to conform the foreground to the background. Since you can reiterate Impressions quickly, you can select individual points, reposition them, and re-conform them to the background very rapidly. Impressions tested very well with LightWave 6.0, or at least the Special Beta Force build 366 I've been running. To install the plug-in, I simply copied the Impressions2_1NoNag.p file from my LW 5.6 C:\NewTek\LightWave3D\Plugins\Modeler directory to my 6.0 beta C:\LightWave\Programs\Plugins\Model directory, then added Impressions via the Add Plug-ins… option of Modeler's Preferences menu. The plug-in worked flawlessly, just as it does in 5.6.

There are only a few steps to the process of using Impressions to reshape a model to fit a form. It's just a matter of repeating these steps, with appropriate artistic judgment to get the results you want.

Load the recycle object, and send the layer to the background. This is the generic head model that you have used to successfully model and animate a character in the past. LightWave 6.0 includes a variety of head models you can use for practice.

Load the form object. This is the Cyberscan, digitizer or other new model, like that in Figure 2, that you want to make compatible with an old model that is efficient to set up and animate.

Move, rotate and scale the form object to fit as closely as possible the orientation and size of the recycle object. The distance between the eyes, and between each eye and the base of the nose, are good reference points for most humanoid heads.

Send the form object layer to the background, and bring the recycle object layer to the foreground. The form object should appear as a black wireframe. Your form and recycle objects are now set up for Impressions.

The next pair of steps describe how to apply the Impressions plug-in to conform the recycle object to the form object. You will need to repeat these steps many times to complete the process, so you should learn how to use hotkeys to make each iteration as quick and easy as possible. Also, you should start with points that are easiest to locate in all three orthographic views. I suggest starting with the points that define the profile of the nose.

Select a point, move it well clear of the form's surface, and align it in the Front view with the final location you want for that point.

Apply the Impressions plug-in with an Arbitrary Direction opposite to your movement of the point away from the form's surface.

If your heads are facing forward, you can make most of these changes by pulling the point -Z, positioning the point in the Front view, and applying Impressions in an Arbitrary Direction move +Z. Think of it as pulling an elastic skin off the form, and letting Impressions snap it back for a close fit. Figure 3 shows the results of Impressions' moving the selected polygons to fit the background form. With practice and care, you can move groups of points with a single operation. Don't get greedy; it takes far more time to fix an error than to do it right with fewer points.

Repeat the preceding two steps until you have reformed the entire recycle object to fit the form object. When you are done, save the foreground layer with a new name.

Once you have a recycled object, you can use Joe Alter's Lip Service or Glenn

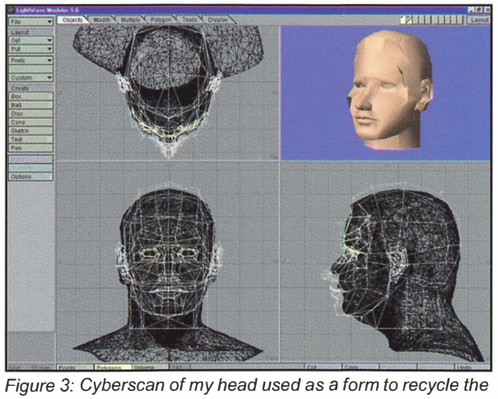


Figure 3: Cyberscan of my head used as a form to recycle the Caucasian_Male model.

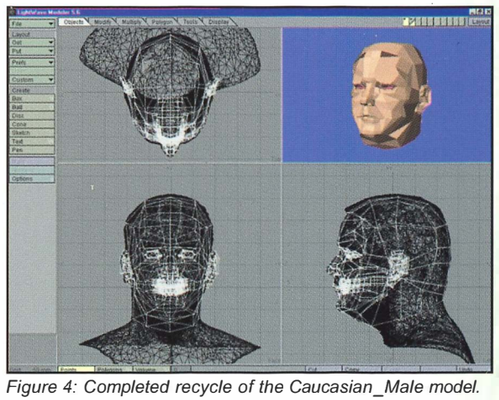


Figure 4: Completed recycle of the Caucasian_Male model.

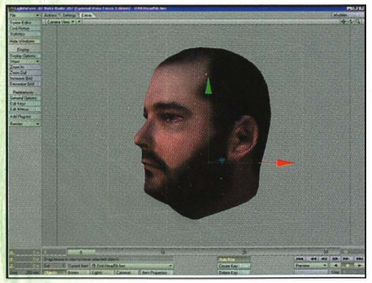


Figure 5: Morph between the Caucasian_Male model and the recycled version of my head, with Cyberscan color map applied.

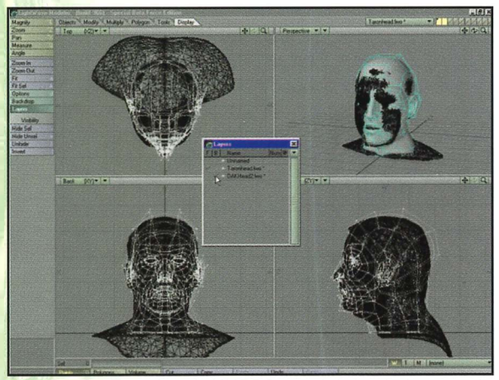


Figure 6: Taronhead NURBS model loaded over the background form of my Cyberscan head, ready for recycling.

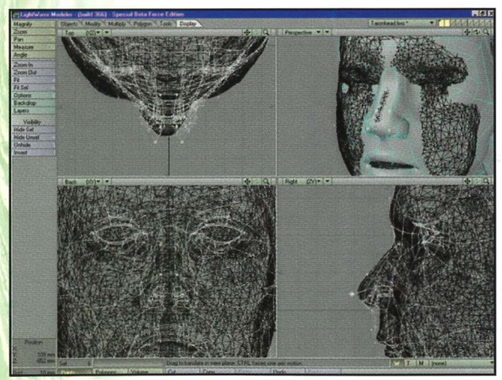


Figure 7: Six NURBS controls moved to match the NURBS surface of the nose closely to the underlying wireframe of the Cyberscan head.

Lewis' Similar software to convert entire morph target libraries to the new shape, as I've described in previous columns. Because the original recycled object and the reshaped one have identical point counts and distribution, all the morphing tools available in LightWave 5.6 will be able to morph them. Figure 5 shows an example of using this morphing capability to apply a color map accurately to a very different head shape. Create your own evil twin!

LightWave 6.0 presents some interesting new modeling and animation tools that can make polygonal models a less attractive option. In these cases, you will want a way to adapt a NURBS model to a form. Impressions won't be much help, because shrink-wrapping the NURBS controls to the form surface will create a NURBS surface that is markedly different from the form surface. Instead, you will need to manually position the NURBS controls to match the NURBS surface to the form surface. Fortunately, a simple setup in Modeler 6.0 makes this a simple and intuitive process.

Load a NURBS model. For this example I'll use the Taronhead object in the L6 Projects\Character\Heads directory of the standard 6.0 installation. Send this model layer to the background.

Load a form object, in this case the Cyberscan of my head, to the foreground layer.

Make sure the Perspective view is set to Smooth Shade. This is important because the background layer will be shown as a black wireframe, and the difference between the wireframe and the smooth shaded foreground will be an important guide for your fitting operations.

Move, rotate and scale the form object to fit as closely as possible the orientation and size of the NURBS surface, NOT the control cage. Again, use the interpupillary distance and the distance from each eye to the base of the nose as reference points.

Change the form object layer to the background, and the NURBS object layer to the foreground. You should now see the NURBS object as smooth shaded in the Perspective view, and the form object as a black wireframe. The wireframe should show through the NURBS object in some places, as shown in Figure 6.

You need to modify the NURBS control cage until the wireframe barely shows through, all over the NURBS object surface. Figure 7 shows six control points adjusted over the upper surface of the nose. Note that the underlying wireframe barely shows through the smooth shaded surface, indicating that the surfaces are closely matched. If you can see all points and edges of the polygons, the NURBS surface is too deep; if you can't see any of the black wireframe, the NURBS surface is too high. If you can see most of the black points and parts of the edges, the NURBS surface is just right.

This particular model has a total of 586 control points to modify, but the control positions for much of the head surface are no-brainers and go very quickly. The finer details around the eyes, nose and mouth require more time and care. These are very small adjustments, so you will probably find it easier to use the numeric requester. If you select a group of control points that should all move in the same direction, you can repeatedly apply a very small increment (a quarter millimeter or less) to get very fine control.

If you have a head model of your own lying around, I highly recommend using one of these methods to either adapt another model to your model's shape, or to adapt your model's mesh to another model's shape. Either way, you will develop a useful (and potentially lucrative) skill for recycling models.

You can get a list of Cyberware service bureaus by writing to Cyberware, 2110 Del Monte Avenue, Monterey, CA 93940, calling 831-657-1489, emailing sue@cyberware.com, or visiting their website at www.cyberware.com.

For more information about Impressions and other plug-ins by Dave Hudek, visit his website at http://www.ultranet.com/~dhudek/graphics/lw/ or email him at Hudeks@AOL.COM.

Doug Kelly is Editor of Keyframe and a freelance writer and animator. You can contact him via email at dakelly@megalink.net or through his website, www.megalink.net/~dakelly.

I hope everyone is having fun playing around with the new radiosity and caustics features in L[6]. Great, a whole bunch of new features to abuse; let the new lens flare controls seduce you. However, for all the old-school slosers out there, I'll continue from last issue, using my "LightWave classic" light theories and perturbations. (Slosers? New slang, please integrate it with your cool friends - if you have any. Well, you have one: me. But please, call me Matt.)

Here's a quick refresher of the guidelines explained in Part I:

- *Try to keep all lights at 255-255-255 White.*
- *If you're going to use colored lighting, color correct your textures to their OG or native color.*
- *Keep your key light, or multiple lights that are aimed most directly at your element, at 100% or totaling 100%.*
- *200% Lights = 100% Light with 200% diffuse and 200% specularity.*
- *Keep Diffuse at 100%! A color of 255-255-255 at 50% diffuse = 128-128-128 @ 100% diffuse.*

Basically, forget everything you ever learned about the Diffuse settings, and ghost it from your vision when you look at the Surface panel. Your goal is that your Photoshop color map and your render look the same, with the exception that the render respects the modeling and is using 3D lighting. I'm not talking about a 100% Ambient effect here; you just want to get the most predictable results. Think of it as a new feature, LightWave rendering interactivity in Photoshop.

To get started, use whatever live-action camera information you have to set your camera's zoom using the Lens Focal Length. Remember that LightWave's readout of Lens Focal Length varies with the Film Size, so if you know your film size, enter it first. If you don't know the film size, take your best guess, or call the DP; they often love to talk. Better yet, invite him/her over to give you lighting advice). If you are compositing a CG character and you had a maquette on location to shoot for lighting reference, take a small vacation for the amount of work you saved yourself. It's a good idea to have a low-res copy of your character, or even a blob in roughly the same size/shape for your lighting experiments. This will also help you to judge scale.

You'll always want to model and texture a small existing area of your plate photography. Always. This is the best way to study and calculate your lights, fog, and texture colors. It's also a great way to line up and test your camera lens settings. Choose objects from your BG plate with curves and edges and a few flat things like the walls and the ground. Make sure you model any geometry that would cast shadows onto your CG elements. In the firepit example, I cheated this by using a spotlight projection image for the heavy shadows that are on both sides of the plate. I don't really recommend this, but it worked well for this scene. Make sure the areas that you choose to model are ones that will be local to your CG. There's no reason to match your lights to a building 100 feet away, if your character is going to be 5 feet from the camera. You should keep your geometry simple unless you are going to be using it to create precise mattes. If you have blueprints or diagrams taken from the set, you'll definitely want to use them. Otherwise, model things as close as you can guess. Try to model in the ballpark of the correct scale, but you'll want to size, rotate and adjust the CG models more interactively in Layout. Make sure when placing your geometry over your plate that you pay attention to your top view. Make sure objects are in correct relation in Z-space before you start sizing them.

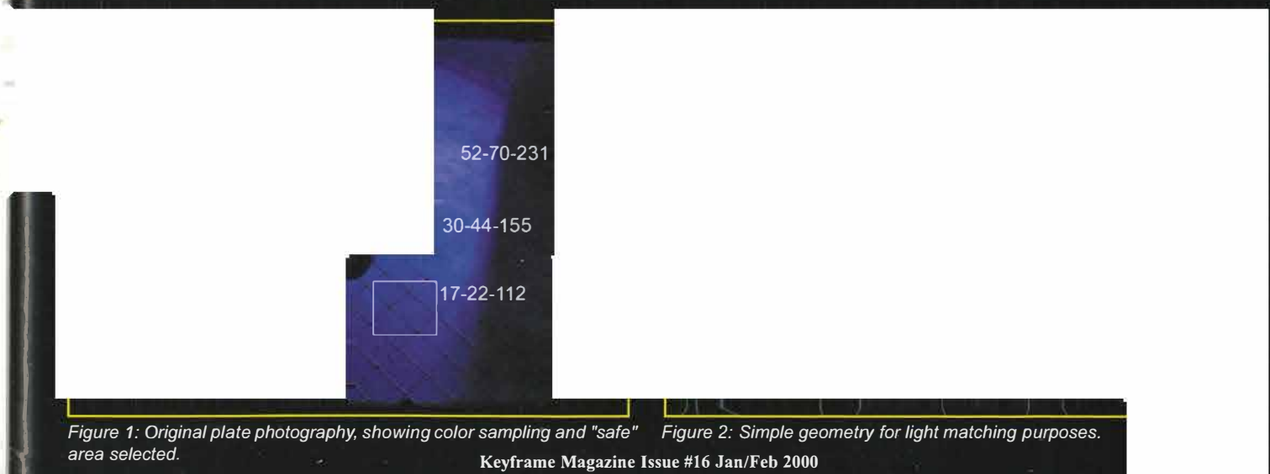


Figure 1: Original plate photography, showing color sampling and "safe" area selected.

Figure 2: Simple geometry for light matching purposes.

For the "firepit" example in Figures 1 and 2, I've modeled the 8-sided firepit with a diameter of 2.7m. This is correct to scale according to its blueprints. Aligning this was a little tricky, because the camera angle is so severe and I didn't have any angle information for the live-action camera. Often you'll get information on the lens angle, height, f-stop and focal distance, but not the camera rotation.

If you try to use this geometry for mattes or highly scrutinized shadow catchers, you'll notice that no matter what you do, your geometry won't line up, even if you run it through 2D lens distortion filters. This is when you stick bones in the model and make it line up. Honest! The most important skill to have in 3D is brute force. I know it sounds like a cool plug-in, but all you need is bare hands and your noggin. The Bone Limited Region setting can help you get precise point control in Layout. This technique works really well for aligning ground planes.

Here's a tip for quickly adding bones to a ground plane: Start with a low-res plane (less than 50 points), copy it into another layer, move it down a bit, and paste another copy. Kill the points, and from the top view, select each point (which will actually select two points, one from each plane) and hit ctrl-p to make a curve. Do this for each point you can see from the top view. When you've added curves for all the points, use the Skeleton Maker plug-in to convert all the curves into a bone scene for importing into Layout. In Layout, make sure you set your bounding box/polygon threshold high enough so you can manipulate those bones in real-time. With your plate displayed in the OpenGL Background, you're all set to start dragging points to distort the ground plane to precisely match the background image.

Figure 3: Simple Texture extracted from plate photography.

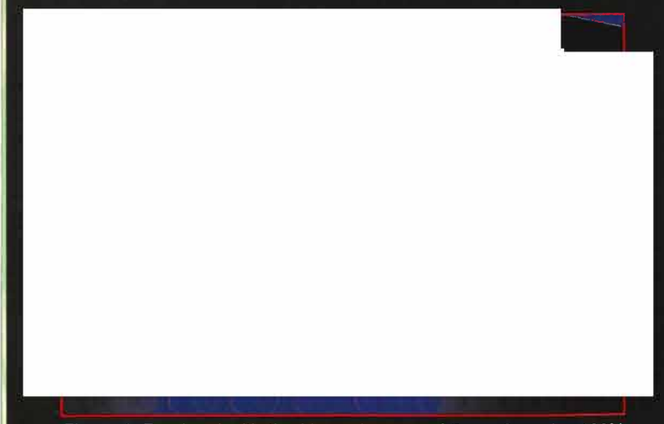


Figure 4: Rendered with the blue color map of figure 3, and a 100% white light.

Next, you'll want to make some dinky texture to map onto your plate matching objects. Load your BG plate in Photoshop, and select the most normalized area for whatever geometry you are mapping. The area you need for your texture map should be devoid of lighting and diffuse effects. Take a look at the ground in Figure 1. There are a lot of shadows at the sides of the plate. A specular highlight is evident in the upper middle portion of the frame. Select the safest area, where the ground is in light, but not too bright or dark. Repeat that area, or do some cheesy cloning/pattern filling. Save your resulting texture (Figure 3) for use as a FPM (Front Projected Map) or whatever. Add it as a color map to the appropriate model surfaces. Do this for whatever BG geometry you model.

In Layout, take a look at where the lights are coming from and place some lights to match what you see. Carefully study your background plate for lighting and shadow hints. Unless you absolutely need Area lights, you should stick with the standard LightWave "classic" lights. Area lights don't subscribe to the same 100% Intensity system and their shadows can take quite a while to render. Guess at some light values/intensities, keeping the 100% intensity rule in mind for the amount of direct light. Looking at the firepit plate, we can see a light coming from the north of frame, shining down at a 35-45 degree angle. Ambient light is very low. Remember that tweaking the light angle with small angle adjustments can give you dramatically different results. Initially, keep all your lights pure white, and don't crank any intensity over 100%.

It's time for a test render. Adjust, F9, adjust, F9. If you're using QV, judge your colors against the BG plate using its color readout. Otherwise, load both your BG plate and your F9 rendering into Photoshop for color and intensity comparisons.

Don't be afraid to use Diffuse levels of 150-200%. I was at first; it scared me. You can use this as a way to adjust your light intensity surface by surface. Refer back to the example of the firepit and the textures we made for it. In the real world, the light is at a 35-degree angle to the ground. When I align my LightWave light at a 35-degree angle to the ground, my texture won't be the same color intensity as the BG plate. It would have the same intensity if the light was pointing directly at the ground, but that's not logical. I really need my color texture to be what the ground would have looked like if a 100% light were hitting it directly. So how do you get LightWave's 100% intensity in the real world? Keep your lights at 100% and adjust the color map's brightness or crank up the Diffuse setting until things match. This keeps my variable values normalized, since whatever CG element I plan on compositing with this plate will have Diffuse values of 100%. My initial test render of the firepit scene was very dark, but instead of cranking up the Diffuse, I gave the surface some Specularity, which seemed more appropriate. Take a look at Figure 4, only my second render. You really have to watch out for specularity and how much it can brighten a surface, especially when using low glossiness settings.

As much as I've preached about not using colored lights, the "firepit" is a good example of when you would want to use them. The light is so evenly blue, and there's also a glowing red fire pit with a high falloff. This is why I made a grayscale version of our cheesy texture. Before I made it grayscale, I noted the intensity range of the hottest color channel, blue. In my texture, the blue channel varied between 85 and 121. After I made it grayscale, however, the highest color value I could get in any of the color channels was 70. This is why I used Photoshop levels to brighten it up until it was in range of my former blue intensity. It's a good idea when using Photoshop to keep your info palette visible at all times; it will always show you before and after color values for whatever operation you are using. This made it easy for me to bring my grays into the correct value range. Now I can light it with a blue light and it should come out the same. I noted through color sampling that the "hottest" colors on the ground were 52-70-231 (see Figure 1). I changed my white fill light to this color. As you can see from my very first render in Figure 5, this got me pretty close. I still have some tweaking to do, but I got to this point for free because my foundation was strong.

I highly recommend compositing your shadows. This is the easiest way to tweak them. You can color them any color you want, adjust their intensity and blur them. This is especially necessary if you are ray-tracing your shadows since they render so sharp. If you're using shadow maps, you gotz da Shadow Fuzziness to fuzzin up yo shadow yo.

When I'm too lazy to composite my shadows, I use spotlights, and split them up to get the right shadow intensity. Initially, I use only two or three lights. Later, I clone the same light a few times and modify the clones to better match the BG plate's lighting. I divide the intensity between the lights depending on their purpose. Often I split up one light into three: one for the diffuse and shadow, one without shadows, and one with either an offset angle or with higher shadow fuzziness. You can also use an offset angle (or use Realitools2's Lighter) to soften the diffuse termination. For example, a distant light at 100%, cloned into two 50%-intensity lights with one light pitched 5 degrees more than the other, will soften the terminator.

It surprises me that this is the first time I've mentioned the beloved Realitools2's Lighter plug-in. It makes photorealism much easier. Using Lighter, you can adjust the diffuse termination surface by surface. You can move the diffuse terminator, as well as adjust its falloff. This is very useful for surfaces like cloth, hair, and dull metals. Lighter also gives you up to 3 specular highlights, exclusion and inclusion of different lights, and diffuse transmission, custom ambient, and custom light intensities per light and surface.

I recently worked on the Korbel Champagne Millennium commercial, and we had to render a CG Earth that was half in sunlight and half in darkness with visible city lights. One problem was: How to get the city lights to be visible only in the dark areas? A two-pass composite, or just use Lighter and include/exclude a cloned light pair, faced 180 degrees from each other? Either would have worked, but I just moved the terminator 180 degrees around. With only one light, the diffuse was flipped for the city light layer, and normal for the earth & cloud layers.

Once you eliminate the unnecessary variables, match lighting all comes down to color sampling and color tweaking. Remember, "sample, tweak, sample, tweak." Keep a pad and pen handy, or keep WordPad open to jot down color values. Sample colors from your plate, then sample your test render. Sample, tweak, sample, tweak. Honest, it's the pathway to photorealism. If you want to experiment with these images, you can download the tutorial files from my website.

This firepit scene was for a film called Beowulf. Figure 6 is a final frame of the Witchmother creature, with over 250 Puppet Master sections, rendered over the BG plate with matched lighting and shadow-catching surfaces.

Matt Morgan is a senior animator at Threshold Digital, an F.X facility in Santa Monica, CA. You can email him at matt@threshold-digital.com. Check out his web site at www.fxartist.com.

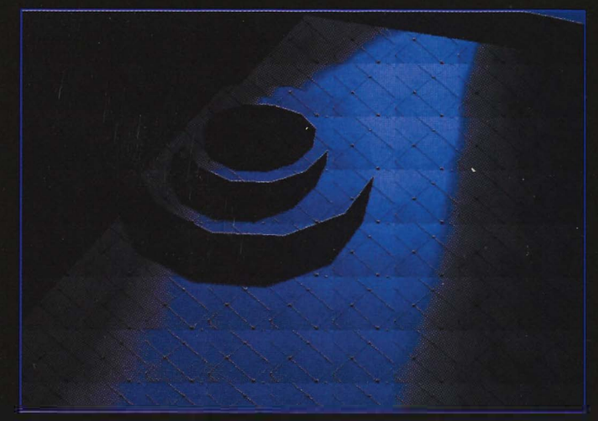


Figure 5: Render with a gray color map and a 100% light colored blue.

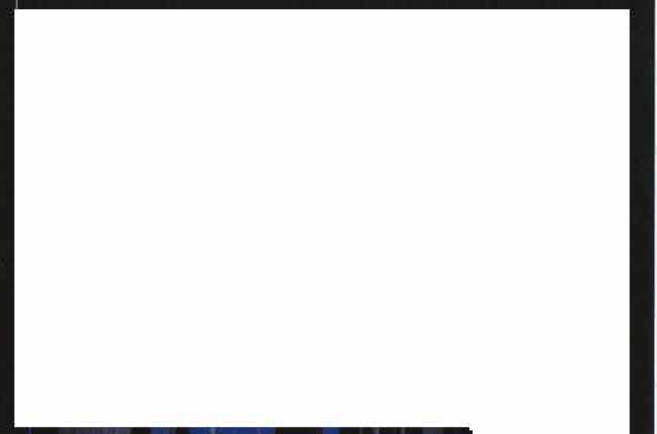


Figure 6: Final Render with integrated CG Creature.

L6, Speed, Illustrator, Thicken, CMYK, Memory

by David Warner

Many people had come to the conclusion that we would not see LightWave v6.0 until some time in 2000. However, NewTek surprised us all with an announcement in mid-December that L6 would be going to a public beta-testing phase before the end of the year. True to their word, many people rang in the New Year by booting up their copy of LightWave 6.0!

So how good is it? Well, the phrase "diamond in the rough" immediately comes to mind. L6 looks good, really good, but it's got a long ways to go before it will be ready for final release to the general public. There are many features that are incomplete or missing and the program does crash fairly often, but that's to be expected with any program that is still being beta-tested. Once you get a feel for what features are still buggy, it becomes fairly easy to tiptoe through the minefield of software-crashing bugs.

Bugs and missing features aside, LightWave 6.0 looks very promising. I'm sure there is extensive coverage of the new features elsewhere in this issue, so I won't get bogged down trying to list everything, but suffice to say L6 is a major improvement over 5.6. All those improvements come with a bit of a price-tag in terms of interface changes, so people should be prepared to spend some time learning the new layout. However, once you become familiar with the changes, your productivity will rise significantly. As an L6 beta-tester, I'm having a tough time switching between LW5.6 and LW6.0, but it's a burden I am happily bearing.

At this point you're probably wondering exactly when LightWave 6.0 will be out of the beta-testing phase and ready for final release. I have no idea what NewTek's plans are, but I hope that they take their time and get the product as stable as possible before releasing it. There are plenty of cool new features to keep everybody happy, so all they have to do now is squash a fairly long list of bugs and it will be something that everyone should be happy with. My guess is that we'll see a final release of LightWave 6.0 just in time for NAB 2000, which takes place in mid-April. Whenever L6 is released, you can be assured that it will be well worth the wait!

Since L6 is still only available as a beta version, we won't tackle any questions about the new features yet, but here is a healthy selection of questions and answers from the LightWave mailing list.

Q I am trying to reduce the render times on a very complicated animation with full ray tracing turned on for shadows, reflections, and refraction. I have tried rendering the scene in separate layers, but it just won't work because there are so many transparent and reflective objects interacting with each other. Is there anything else I can do to reduce the render times on a scene like this?

A Dave Jerrard, a long-time member of the LightWave mailing list, has come up with an excellent list of things anyone can do to reduce render times when heavy ray-tracing is involved. Here are a number of tips that can significantly reduce render times for any images that are being ray-traced.

1. If you have any null objects in the scene, be sure to turn off all shadow-casting options for them because even though null objects do not contain any polygons, they are still checked for shadows unless you turn off those options.

2. If there are multiple objects that are stationary, relative to each other, then import them into Modeler and save them as a single object. It is much easier for LightWave to calculate the shadow of a single object with the Self-Shadow option turned on than it is to calculate shadows for multiple separate objects.

3. Objects that are not receiving shadows from other objects should not have the Receive Shadows option turned on.

4. Lights that do not cast noticeable shadows should just have their shadow casting options turned off.

5. Objects that have procedural bump maps applied to their surfaces should have low Frequency settings, especially if the objects are far away from the camera.

6. In the Render panel, lower the Ray Recursion Limit value. This controls the number of times light will "bounce" when calculating a reflection. The default is 16. You can reduce this number significantly without noticing any real change in your image, but the render times will definitely be reduced.

7. In areas where you can't see a real object that is being reflected, you should substitute an image map instead. This can be very handy if you have objects out of the camera's view that are being reflected in on-screen objects. Simply render an image map of the reflected objects and map it onto a flat plane that is positioned appropriately to cast an accurate reflection of the original object.

Q I'm trying to import an Adobe Illustrator file into Modeler, but I always end up with solid shapes. For example, if I import a word with the letter "O" in it, the Illustrator Import plug-in creates a solid circle without the center of the "O" cut out. How can I fix this?

A If you take a close look at the Illustrator Import plug-in interface, you'll notice an option called "Auto Axis Drill" with a checkbox next to it. If you turn this option on and then import your Illustrator file, the object will be created correctly with the hollow areas where they should be.

Q I'm trying to reproduce a shiny metal surface effect where an object, like a car, has a very bright and very large specular "glow" to it. How can I create this effect?

A The effect you are referring to is called a "Specular Bloom" and there are a few different ways to achieve this effect. Many people use a plug-in like Gaffer or RealiTools to add the effect to their image. While this is definitely the easiest way to create this effect, there are some simple techniques you can apply that do not require you to purchase a plug-in.

A very effective way of creating a specular bloom effect without a plug-in is to take the polygons in the object you are applying the bloom to, copy them into a new layer in Modeler, and apply a very small Smooth Shift value to all of them. Change the surface name for these polygons and then save the two layers together as a new version of your object. In Layout's Surface panel, you would then select the polygons you created with the Smooth Shift operation and make the Surface Color totally black with a very high Specularity value. Then turn on the Additive function for this surface so that these polygons will be totally invisible except for the areas

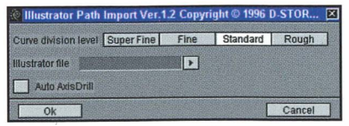


Figure 1: The Adobe Illustrator import plug-in interface.

where the specular bloom occurs. This technique works very well but it can be tricky to get the precise effect you're looking for.

An excellent tutorial on various lighting effects, and specifically for creating a specular bloom effect, can be found on the web for free by going to http://www.tv3d.com/TextTutorials.html and checking out the Studio Lighting Setup tutorial written by Eki Halkka.

Q I modeled an object that has an interior and exterior, but after I was done modeling and surfacing the object, I realized that the division between those two surfaces just wasn't big enough. Is there an easy way to adjust the thickness of an object or do I need to start all over?

A Since you modeled the interior and exterior parts of your object with separate surface names, it will be very easy to adjust the object. Just use the 'w' key (statistics option) to select all of the polygons with one surface name, preferably the exterior surface, and then use the Smooth Shift function to push the polygons outward by an appropriate amount. This would work just as well for the interior surface polygons if you wanted to decrease the interior volume of the object as you thickened up its shape.

It should also be pointed out that the Smooth Shift function could be used in reverse to make an object thinner. Just do the exact same thing as described above, but use a negative Smooth Shift value and the polygons will pull closer to each other.

Q I'm doing some print work and need to generate an image with a 300dpi resolution and CMYK colors. How can this be done with LightWave?

A This is a situation where LightWave needs a little help from another program to get the job done. If you are doing any sort of print work, you definitely need a program like Photoshop to adjust rendered images created in LightWave, and even to help set things up before you start working in LightWave.

You cannot directly create a 300dpi (dots per inch) image in LightWave, but with a program like Photoshop, you can use the New function to determine what the resolution of your image should be. Simply enter the dimensions of the image, the dpi value, and Photoshop will tell you the exact resolution of the image. Then you can use these values in LightWave's Camera panel to define a Custom Size image resolution. Once you render the image, you will need to import it into Photoshop and adjust the dpi setting to 300, because the default value for any images rendered out of LightWave is 72dpi.

The CMYK color values are a little trickier to work with because color values in LightWave are dependent upon a lot of things, like lighting, diffusivity, luminosity, specularity, and pretty much everything that gives an object 3-dimensional depth in your virtual LightWave world. Again, this is a situation where Photoshop is really crucial to get the job done. You cannot enter CMYK color values into the LightWave Surface Color settings, but it's fairly easy to use the Photoshop color picker tool to enter the CMYK color value you want and then transfer the resulting RGB color values into the LightWave Surface Color values.

As I mentioned though, even if you enter RGB color values that are comparable to a CMYK color, you may not get those precise color settings in the final rendering because of all the factors that go into shading an object. You could ensure that the CMYK color values stayed accurate by giving the object a Luminosity of 100% and remove all other surface attributes, but then your 3-dimensional object would not have much depth to it. In this case the best solution is to start with the right RGB color values (as described in the previous paragraph) and render out an RGB image. Then simply load the object into Photoshop and change its color mode from RGB to CMYK. This will adjust all the color values in the image so that they may be reproduced on a printer. If you are doing any sort of print work professionally, then you will quickly find that Photoshop and LightWave complement each other very nicely and should be key components of any digital toolbox.

Q I've heard from a number of people that I can improve my computer's performance by putting the virtual memory pagefile on a hard drive partition that is separate from the partition where Windows is installed. My computer frequently uses virtual memory when I load big scene files into LightWave or when I'm working with Photoshop or Aura and I want to do everything possible to speed up my computer's performance.

A The number one way to improve system performance with LightWave or any other memory-intensive application is to add more memory to your system. Virtual memory is significantly slower than standard memory, so the less time the computer spends reading and writing data to the hard drive, the faster your computer will perform.

Regardless of how much memory you have in your system, you will still probably hit virtual memory from time to time, especially when using programs like Aura and Photoshop, so here are some simple steps you can take to significantly improve system performance.

The virtual memory pagefile should definitely be stored in a hard drive partition by itself. Ideally, you want to create a good-sized partition at the beginning or outer edge of your drive, which is the fastest part of a hard disk. This partition should definitely be on a drive that is separate from the drive that Windows and your other applications are stored on, so the drive head does not have to jump around the platter trying to read from multiple partitions. You should also use the fastest drive in your system for the virtual memory pagefile, preferably a high-speed A/V drive if you have one available.

This "trick" can really speed up performance in operating systems that, by default, have a virtual memory pagefile size that adjusts itself "on the fly" as in Windows 95/98. The reason this is so inefficient is that as data is written and re-written to a drive, the sectors on the drive become fragmented and the virtual memory pagefile gets scattered all over the drive. This can have a very detrimental effect on your system's performance and, depending upon how much virtual memory LightWave is using, your render times can end up being much higher than they should be. Setting up a virtual memory pagefile in a partition on the outer edge of a very fast hard drive can make a huge difference in LightWave's render times, but it will also greatly enhance the performance and reliability of your system overall.

If you decide that this is something you want to do, but you don't know how to drop a new partition into your drive without having to reformat the whole thing, I strongly recommend a program called PartitionMagic by PowerQuest. This Windows utility enables you to resize and move partitions on a hard drive without having to re-format the drive. It also gives you the ability to position the partitions exactly where you want them.

A final note for this issue: The LightWave community recently lost a good friend when Elliot Bain (a.k.a. "OldGeek") passed away on December 15th, 1999. Elliot was a member of the LightWave Mailing List and #LightWave IRC channel for many years and he will be sorely missed. If you would like pay your respects or read what others had to say about Elliot, visit the Elliot Bain memorial web-site at http://216.63.2.24/og.htm. My deepest condolences to his family and friends.

For more info about David Warner or TutorVision 3-D CD, his series of multimedia tutorial CD-ROMs for LightWave, please visit http://www.tv3d.com or send some e-mail directly to Dave at dwarner@tv3d.com. The newest release, Volume 3, contains more than 6 and a half HOURS of video and a total of 26 different lessons for LightWave!

Figure 2: The Smooth Shift Tool

Up Close and Personal

by Jason Goodman

For this issue of *Keyframe*, I would like to discuss a topic that is not LightWave 3D, Aura or the Video Toaster, but should be of interest to users of any of these. I recently have been shooting some exciting footage in stereoscopic 3D. The Nu-View 3D video camera adapter from 3D Video, Inc is an ingenious device that brings the power of stereoscopic cinematography to the average user. It is a small contraption that mounts on the front of most any video camera and takes excellent quality 3D videos.

Previously, shooting motion video in 3D required specialized cameras and equipment that cost tens of thousands of dollars. These systems frequently required certified operators and technicians. They were complex and expensive. Nu-View is only $499. It is compact, efficient and cheap! It makes elaborate and expensive equipment obsolete. It seems like something NewTek would have come up with, doesn't it? The quality of the stereo images taken by the Nu-View lens attachment is surprisingly good. When combined with a digital video camera the results are optimal. The higher resolution and reduced noise of the DV format greatly enhances the realism of the imagery.

Of course the taking lens is only part of the 3D system. Nu-View also comes with special Nu-Shades glasses for viewing 3D videos on a standard TV. A small transmitter hooks up to the video output of your TV or VCR. The transmitter sends signals to the glasses that cause their LCD lenses to open and close revealing each field of video only to its intended eye.

The NuView comes with 2 pairs of NuShades glasses for viewing on an ordinary TV.

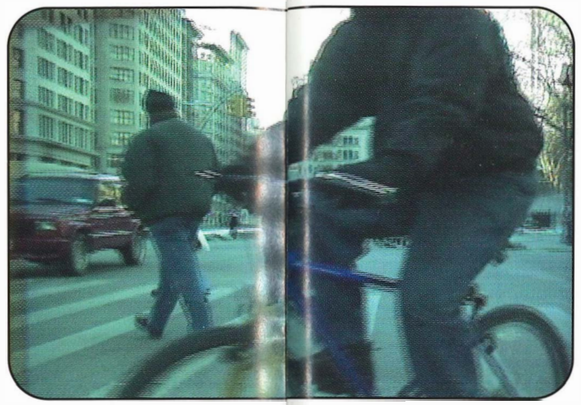
There are several systems for viewing 3D imagery, and there are pros and cons to each. Most people are familiar with the anaglyphic red and blue process used by everyone from Freddie Kruger to Burger King. The anaglyphic process is inexpensive and relatively simple to produce. It is easy to present and it offers a generally good 3D effect. Because of the color disparity between each eye and resulting undesirable effect, true 3D enthusiasts have always held the anaglyphic process in the lowest regard. Many people have also seen polarized 3D films at theme parks or high-end theaters. The advantage here is that the glasses appear to be clear, without the color disparity of anaglyphic systems. Polarized 3D projection can look wonderful, but often requires more complex presentation equipment. It generally does not work in print or on CRT monitors. More recently, electronic LCD shutter glasses have become popular. High-resolution video and computer monitors have given rise to a system that interlaces the left and right eye views in the even and odd fields of a video signal. This is the system used by the Nu-Shades included with the Nu-View adapter. This creates a substantial 3D effect, but I find the flicker of electronic shutters opening and closing sixty times per second very distracting. I don't see this as a limitation of the Nu-View, however.

As a taking system, the Nu-View is a breakthrough. For viewing, one of the best quality systems I have seen uses video projection and polarization. This allows for a larger audience, lightweight clear glasses and a flicker free full color 3D image. 3-D Video, Inc offers digital 3D theaters featuring Emmy Award winning Digital Projection displays. Videos taken with Nu-View are also compatible with various head-mounted displays and 3D projection equipment from other manufacturers.

One of the most impressive features of the Nu-View is its precise control of convergence. When photographing the world in 3D one of the most important considerations is the point at which the left eye and right eye views converge. A small dial on the lens adapter allows you to adjust the point of convergence from approximately one meter on to infinity. Although it would be nice to have some sort of gauge or marker on this control, the point of convergence is easily seen in the viewfinder of your camcorder. You actually see a double image in the viewfinder while you are shooting. When items in the frame appear perfectly aligned with one another, they are at the point of convergence. This is about as easy to adjust as focus and equally important to the shot. I was easily able to determine the distance to convergence by measuring the distance from the camera lens to the subject.

I highly recommend using the Nu-View with a digital video camera. Once you have shot your footage, you can edit it in any standard package, Speed Razor on the Toaster, Avid or Media 100, Premiere, Final Cut Pro, EditDV or whatever you like. Transitions and effects that change the shape, position or scaling of an image or modify the fields in any way may break the 3D effect. Standard wipes, dissolves, fades and cuts should all work without a hitch. If you really want to get tricky you can do compositing in 3D as well. If you are going to attempt anything beyond the most basic composite, I recommend that you first separate the left and right eye views and work on each independently.

When you are ready to output to tape, or whatever your final output medium will be, you can re-interlace the fields for viewing with LCD shutter glasses. I was able to output independent, deinterlaced views by importing the original clip via FireWire, and using Media Cleaner Pro to export it with a deinterlacing filter. I first exported with the "even" option selected. This gave me a full frame movie (interpolated from the even video field) of just the left eye view. I then exported the same clip with the "odd" field setting to create a video of just the right eye view. This can be done in any application that allows you to selectively deinterlace your videos. Premiere does have a deinterlacing option, but it is either on or off, you cannot choose which field you get. A tricky little thing you can do is to copy the Photoshop deinterlace filter into the Premiere plug-ins folder. The filter will work in Premiere exactly as it does in Photoshop; it will allow you to choose even or odd video fields for export. This technique works on both Mac and PC versions of Premiere.



It may be hard to see in print, but this is a frame grab of the interlaced 3D format.

The concept of 3D desktop video and all of the effects that go along with it is exciting. Conceivably you could shoot some footage with the Nu-View against blue screen, and then use a tool like Ultimatte to do some flying out of the screen compositing effects.

Another good use for the separated fields technique is compositing in LightWave. Once you have the independent left and right eye videos, you can output an image sequence (I always recommend .iff format) and bring those into LightWave Layout as background images. With the live action footage in place you can create animation that interacts with your stereoscopic environment. This is much more complex than compositing for 2D playback.

It is also important to record all aspects of your shooting conditions. Take note of the camera's height, distance from subject, distance to convergence, focal length and anything else you can think of. To composite 3D animation with live action 3D you will have to recreate the scene as accurately as possible within LightWave. *Do not forget the interocular*: This is the distance between the lenses of the 3D camera. The interocular for the Nu-View is 2.66 inches. This number can easily be sent in LightWave 5.6 or 6.0's stereo rendering menu.

Alpha Channel

A new feature in 6.0 is the ability to have multiple

you don't have 6.0 you could also just create two different scenes for your left and right eye views. The reason I recommend two cameras or two scenes rather than using LightWave's stereo compose features are due to the limitations of that plug-in. Stereo composer is really only designed to create anaglyphic (red/blue) compatible scenes, and it has problems when used in cross platform network renders. If you have a full frame left eye movie and a full frame right eye movie you can combine them in any way you like. This gives you the versatility of creating one original animation that can be viewed at your web site with red and blue glasses, on your home television with LCD shutter glasses, and in a digital projection theater with clear polarizing glasses. If you do not own After Effects, Aura or a similar tool, you can combine the left and right eye full frames into Nu-Shades compatible field interlaced video in LightWave. Create an alpha channel that alternates each horizontal row of pixels with black and white lines. Use this as the alpha channel with your left eye images in the foreground and your right eye images in the background. The newly rendered LightWave frames should combine the two into a single interlaced sequence.

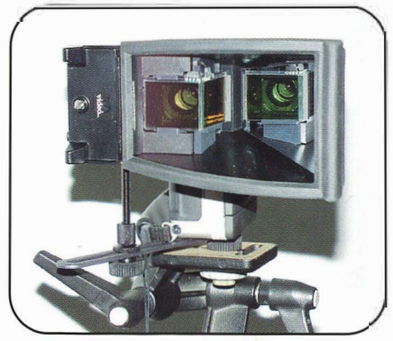
The Nu-View video camera adapter is a wonderful device, but it is not for everybody. It adds bulk to your camcorder, and it definitely draws attention while in use. Not all videos are going to be enhanced by being in 3D either. You may find it annoying to watch a two-hour video of someone's wedding in 3D (or 2D for that matter.) But if you miss the 50's or have a hankering to make the next Friday the 13th or Jaws sequel, the Nu-View may be just the ticket. Higher end users should be on the lookout for a "pro" version, which has been announced on the 3D Video web page.

Sidebar: Video Toaster NT

Those of you who have been following this column have probably come to realize that I am not the type who conforms to standards. Frequently these pages have veered from the path that the "Macintosh Up Close and Personal" title might dictate. Perhaps it should be called Jason Goodman Up Close and Personal, because that is the way I have always approached it. I have been involved with computer technology and NewTek products for about fourteen years. These years of experience have given me some knowledge of animation, video, graphics and interactive media. It is my desire to share that knowledge and experience with *Keyframe* readers. While the Macintosh is an important part of the computer graphics world and my production studio, it is not all things to all people, including me. It is therefore often necessary for me to seek solutions outside the realm of the Mac OS, and in turn, report on them here. As with anything, some people will love what I have to say and some will hate it. That, as they say, is what makes horse races.

In my last column, I reported on a new development that I, along with many other *Keyframe* readers, had eagerly anticipated. That development was the Video Toaster NT. In my "review" of the product I pointed out several things that I saw as problems. This really ruffled some feathers and I would like to address that issue. The first thing I would like to do is to apologize to all of the people at NewTek who felt that the writing was unfair. Many people at NewTek have worked hard for years to bring us innovative products. NewTek personnel have also worked closely with my own company on many occasions. By testing the Video Toaster in an unapproved configuration I introduced a problem that is not inherent to the system. It was irresponsible of me to bring attention to the issue that arose from this. When a user follows NewTek's guidelines for approved systems and components, they will not see this problem. I do however stand by my assessment of the Video Toaster in its current revision. I will not reiterate that stance here, but I do encourage the curious to read and evaluate the column for themselves. Take a look at NewTek's approved hardware list http://www.newtek.com/products/faq/1.html if you are planning on setting up a Video Toaster NT suite.

Jason Goodman is the CEO of NYC based Multimedia Software Design. You can email him at jason@msd-nyc.com or visit their website, www.msd-nyc.com.



Interior look at the NuView.

The NuView does add bulk to ultra-miniature cameras, but considering what it does, it is very compact.

Image created by DigiMania, a member of Digital Animation Group PLC, and ©1998 Eidos Interactive and ©1998 Paramount Pictures.

DigiMania, a member of the Scotland-based CG animation company, Digital Animations Group PLC, has a growing reputation for producing some of the most highly acclaimed CG sequences for commercials, television and video games. Based just outside Glasgow, DigiMania is home to about 30 people. The studio has been using LightWave 3D for all its productions since NewTek ported the software to the PC.

DigiMania was contracted by Eidos Interactive to produce all the CG video sequences for their Braveheart *video game. This article will cover some of the techniques used to create "William Wallace," the lead character in the game, based on the film character as played by Mel Gibson.*

Modeling

Press images from the *Braveheart* movie and images from the web were used as reference to build the head. The first stage was to figure out how we wanted the polygons in the face to flow. All vertices and polygons were lovingly handcrafted with the artist constantly tabbing between polygon and MetaNURBS mode, with the results shown in Figure 1. The number of vertices were kept to a bare minimum, which made it easier to drag the points around, control the curvature of the facial features, and create morph targets.

The low-res polygon cage looks reasonably like Mel Gibson, but with the patch subdivision set to about 2 and the model frozen he looks much healthier, as shown in Figure 2. At this stage we had the base pose from which we could derive all morph targets.

The next stage was to start texturing the model. We used Ernie Wright's Unwrap plug-in to cylindrically unwrap the mesh around the Y-axis. This produced a bitmap image, Figure 3. This bitmap was then loaded into Photoshop as a template to start painting over, safe in the knowledge that the map will fit the mesh perfectly when it is mapped back on in Layout.

Texturing

Textures were layered up in multiple channels to achieve the final look. The color map, Figure 4, was the first one created. Parts of this map were pasted in from photographs; the remainder was painted in. The unwrapped wireframe template shown in Figure 3 was used at all times as a positioning guide.

We used a second color channel, Figure 5, on William Wallace for all the battle scenes when he needed to have his face paint on.

The next map was the bump map, Figure 6 on the next page. This was used to add all the fine detail that it would be impractical to model. Skin pores, stubble, eyebrows, creases and wrinkles were all added this way.

The final map is the specularity map, Figure 7 next page. This was used to control the way that light reflects from the skin. You can see from the image map that the forehead and nose are quite white, and will therefore produce specular highlights much more readily than the cheeks, which are relatively dark.

Rendering

Before rendering could take place the hair, clothing and the rest of the body had to be added. As shown in Figure 8, the model was posed for a high-resolution print that would be used in marketing the game. Fiber Factory 2 was used for the hair. Key strands were placed in Modeler, and FF2 cloned them up at render time to create denser hair.

Leave to simmer gently on the render farm for several hours and voilá, one Scottish Light-Wave Warrior as seen on the cover.

The other images that accompany this article were all rendered at high resolution for use in the game's marketing campaign, and appeared in magazines, ads, posters, and the box packaging.

And Finally

We had some fun on the *Braveheart* project. One of the most memorable moments had to be when we got a call from the boys in marketing, asking if we could re-create the shot from the movie when the Scots all turn their backs on the English army, bend over and lift their kilts to expose their hairy Scottish arses. Let's just say we hadn't bothered modeling the Scots quite that accurately, and there was a distinct lack of polygons up their kilts. Needless to say, that shot never got rendered. However, we were faithful to our Scottish heritage in maintaining the tradition that a Scotsman wears nothing under his kilt, even if we did take it a little too literally.

Images created by DigiMania, a member of Digital Animation Group PLC, and © 1998 Eidos Interactive and © 1998 Paramount Pictures.

Laurie McCulloch is Development Director at Digital Animations Group PLC, Hamilton House, Phoenix Crescent, Strathclyde Business Park, Bellshill, Lanarkshire, ML4 3NJ, UK. Telephone: +44 (0) 1698 50 33 00, fax: +44 (0) 1698 50 33 99. You can e-mail laurie@digital-animations.com or visit their websites at www.digital-animations.com and www.digimania.com.

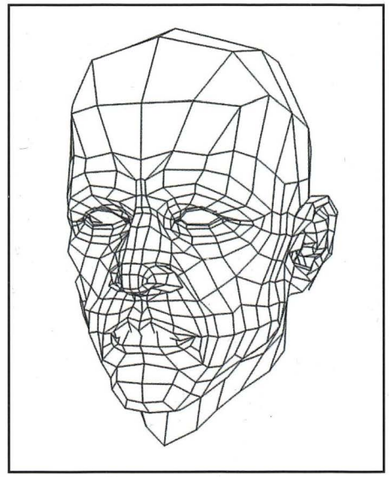


Figure 1: The hand-modeled low-res polygon cage for William Wallace's head.

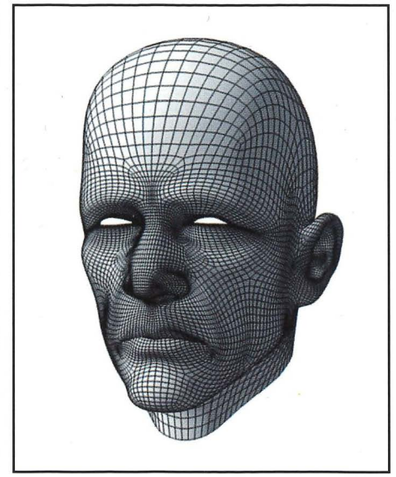


Figure 2: Low-res polygon cage after a patch subdivision of 2.

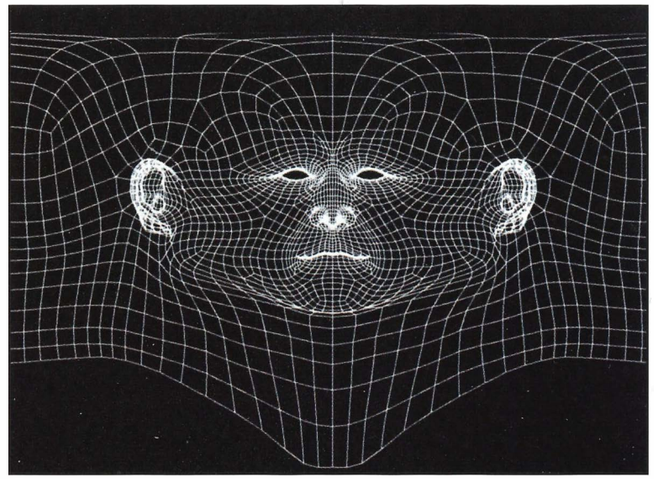


Figure 3: Bitmap template produced with Unwrap plug-in.

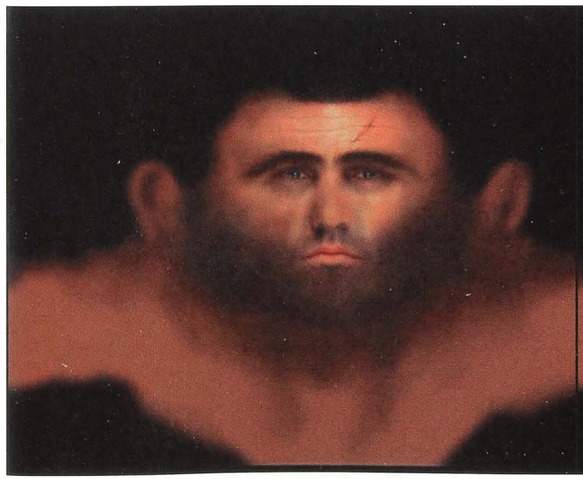


Figure 4: First color map for William Wallace head.

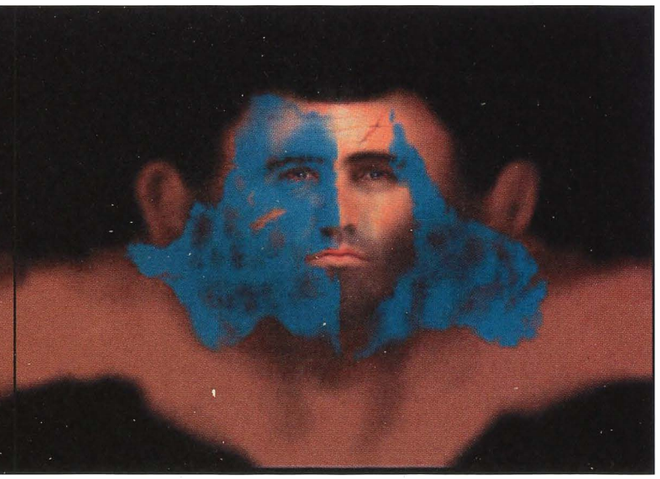


Figure 5: William Wallace battle scene color map.

Figure 6: This bump map was used to add fine details.

wireframe posed for marketing print, added using Fiber Factory

Genesis V2 Pro
A collection of high-end visual effects.
Genesis V2 Pro gives you complete and infinite control over any kind of optical image you can create.
Free Upgrade to LW [6] from Future Fantastic!.....**295.00**

Canopus DVRex-M1/Premiere -
Combines high-quality DV video with multi-channel audio support; includes Premiere 5.1, Kick Start Training CD, breakout box, Rex Edit 2.5, and cables.....**2399.00**

THE JOY OF SIX!
LightWave [6] – Experience the seduction of **LightWave [6]** ! Completely redesigned and mastered, with the artist in mind, for creating exceptional realism faster and easier than ever before! Powerful new tools include enhanced interactivity and scene editor, Skelegons, MetaNURBS in layout with point weighting, caustics, UV mapping, surface editing in modeler and so much more! LightWave 3D is the most used animation system on the planet. What are **you** using?.....**2250.00**

LightWave 3D Competitive Upgrades.....**Call**
LightWave 3D Upgrades.....**Call**
LightWave VT – [6] Upgrade.....**Call**
Aura.....**Call**
Aura 2.0.....**895.00**
Aura 1.0-2.0 Upgrade.....**295.00**
Inspire 3D.....**389.00**
Video Toaster NT.....**2995.00**
NewTek
Visit: **www.videotoasternt.com**
Complete reliable, quality turnkey Toaster workstations are available.

Mega Production Bundle -
All-in-one super package includes DVRex-M1 breakout box, Rex RT option, Rex Edit 2.7, Adobe Premiere 5.1, Rex Navigator software, and all cables.....**2999.99**

Life Forms 3.5 LE –
The latest version of the defacto standard for creating realistic figure movement for 3D animation. Its intuitive interface and powerful features will help you save time and be more productive than ever. Our new entry-level package gives you all the features of Studio without the application support of 3D content. Great for creating Quicktime movies, VRML output for the Web, input for Strata Studio or trueSpace.....**139.00**

Life Forms Studio 3.5 –
All the great features of LE, plus support for many of today's top 3D animation packages. The Studio version includes PowerMoves 1 & 2, a collection of over 750 motion files (valued at up to $5,000) to get you moving fast!.....**432.00**
CREDO The Character Motion Company

formats.
ReelMotion is a realtime simulator that uses physics and collision detection to create realistic animations of objects and vehicles including cars, aircraft, motorcycles, helicopters and nearly any rigid body object.....**634.99**

Oxygen VX1
Achieve true 2D and 3D superb graphics with the GLINT R3 processor.....**219.00**
Oxygen GVX1.....**699.00**

PERMEDIA3 *Create!*
Permedia3 Create! AGP
Deliver vibrant 32 bit graphics with 100% Open GL acceleration for Windows95/98, as well as Windows NT. Permedia3 Create allows the user to achieve high resolution as well as maximum performance on any software application. Included are two valuable software packages, Cyberlink PowerDVD for full featured DVD playback and Colorific, which calibrates your screen for display and printing consistency.....**139.00**
3Dlabs

LIGHTWAVE PLUG-IN/ LIGHTWAVE RELATED SOFTWARE

3D Super Bundle (Art Recon)	539.00
Alpha Magic	199.00
Amazing Sci-Fi Textures	129.99
Animation Master 2000	279.99
Animation Master/Simbiont Bundle	319.99
Aqua Pack	179.00
Bryce 4	179.99
Complete Effects Library	389.00
DarkTree Textures	289.00
DirtyUp	182.99
Enlightment (electronic version)	79.00
Hollywood FX	Call
Light ROM Super Bundles	89.00
Lip Service	269.00
MacroForm 2.0, Intel	289.00
Motion Designer2	399.99
NatureFX	699.00
PaintProps (Aura) (electronic v.)	69.95
Particle Storm 2.0	CALL
Particle Storm SE to Pro Upgrade (for LW[6]!)	~~399.00~~
Power CG Plus/Speed Razor	~~259.00~~
Puppet Master	~~299.00~~
Relativity	
Seamless Textures 4 – Classic Stone	
Seamless Textures 5 – Downtown Surfaces	129.00
SecondFX	189.00
Shave And A Haircut	95.00
SurfacesROM	36.00
TileMotion	535.00
Tree Druid, Intel/Mac	169.00
Virtual Trees & Foliage	169.00
Visual Effects CD Set	140.00

dpsReality
The world's first Studio Digital Disk Recorder! Features compressed or uncompressed recording, alpha channel, balanced/unbalanced stereo audio I/O and DPS Virtual Tape File System for group rendering. The package contains the dpsReality player application, Digital Fusion DFX compositing software, and an audio/video breakout cable.....**2249.00**
Reality *Studio Digital Disk Recorder*
DPS

dpsVelocity – 2D
The combination of hardware and software is what makes this real-time NLE system the ultimate choice. The **dpsVelocity – 2D** delivers faster, sharper, and easier edits at an affordable price. This real-time editing solution includes the dpsReality card, dpsReality player, audio/video breakout box cable, dpsVelocity NLE software, and so much more.....**2699.00**
dpsVelocity FASTER, SHARPER, EASIER EDITS
dpsVelocity – 3D.....**4499.00**
dpsVelocity NLE Software.....**805.00**
dpsVelocity – 3D Bundle.....**5349.00**

Information: 262-548-8120
Fax 262-548-8130
Tech Support / RMAs:
262-548-8159, 12-4 pm Mon.-Fri.
Safe Harbor Computers
W226 N900 Eastmound Dr.,
Waukesha, WI 53186

YOUR NUMBER 1 SOURCE FOR DIGITAL VIDEO, ANIMATIO

The Versatile Hand: Bones & Expressions in Project: Messiah

by John Riggs

Hands are probably one of the most difficult body parts to set up correctly in LightWave. You can end up with fingers that bend like a garden hose, and that looks funny. If you put in enough bones to get it to look right, how are you going to animate it? Are you going to keyframe each bone of each finger manually? That would take forever! There are other options, like Prem's Relativity expressions plug-in, which enable you to use nulls to rotate the bones in the hand. You could even use PhaseLink, one of the Polk plug-ins. However, we are going to do it all in the new plug-in from Station X Studios called Project:Messiah (P:M for short).

The goal of this tutorial is to show you one way of setting up a human hand so it looks good and is fast and easy to animate. The first part of this tutorial explains how to add the skeletal hierarchy of bones to a hand object in Project:Messiah. The reason I am going to use this plug-in is because its bones are a heck of a lot easier to set up than LightWave's bones, and they work better when bending joints. The second part of this tutorial explains how to set up the expressions needed to animate the hand efficiently. We will be using the Slider object in P:M and expressions to animate the fingers individually or as a group. We will also add a slider to turn off the Sliders, enabling you to animate the old fashioned FK way.

Figure 1: Project:Messiah interface with hand object loaded, and the bone in Move mode.

Boning The Hand

Start LightWave, then start P:M. To load the hand object, choose the File|Load menu option. P:M uses a smart loader, so just go to your objects directory and pick your hand object. I built the hand shown in these figures just for this tutorial. Later on, you will have the option to replace the geometry and retain all the bones and expressions.

You need to tell P:M that you want to add a skeleton to the object. Make sure you have the Hand object selected and then click the Effect Tab. Make sure that next to the Effect pulldown it says "Bone_Deform". Click on the Add Effect button. This enables you to add bones to the hand. Click on the Animate Tab, and then make sure the Setup sub-tab is selected. Now you are going to add the first bone. Click on the "Add Child" button. Your screen should look like Figure 1.

You need to place the bone in the correct position in the hand. First make sure you are in Move mode. You can do this just by looking at the bone's handle. If it has an M in the middle of it then it is ready to move. Other symbols it can have are R for rotate and S for scale. Move the bone into position. Make sure to position the bone in all three dimensions (X, Y, Z).

I always try to use real world objects for reference, so I look at my own hand for information on where to put the bone. Take care to notice where the bone ends, because a lot of people place the first bone in the hand incorrectly. The bone should end before you get to the point in your finger where they separate, Figure 2. You will notice that I also rotated and stretched the bone. I rotated it by clicking on the H on the sphere, and aligned it correctly. To adjust the length, just click on the little blue bull's-eye at the end of the bone.

Figure 2, above: Bone positioned, rotated and stretched.

Next you need to add another bone for the finger. Click on "Add Child" again. Rotate and adjust length for that bone also. Your results should look like Figure 3. Repeat the Add Child, rotate and stretch operations until you have a total of four bones as shown in Figure 4.

Figure 3: The second child bone has been added, rotated and stretched.

Next you need to name the bones. Click on the File Tab. Click on the Name Tab. Select the bone Skeleton_Bone_1. In the name field, type "PointerFinger_MAIN" and then press the Enter key. Make sure you press the Enter key because P:M will not update the field you are in unless you do! Name the next closest bone PointerFinger_01, then PointerFinger_02, and finally PointerFinger_03.

Now you need to clone that finger and set it up. While still in the File Tab, click on the Clone sub-tab. Select the bone named "PointerFinger_MAIN" and click on the Clone button. Now move the new bone chain over to align it with the next finger in the hand object.

finger have been added, rotated & stretched.

Figure 5: The MiddleFinger bones have been cloned, rotated and stretched.

Next, rename the cloned bone chain to reflect which finger it is. Click on the Name sub-tab. Select "PointerFinger_MAIN_1" and change the name to "MiddleFinger_MAIN". Rename the rest of the bones to "MiddleFinger_01", "MiddleFinger_02", and "MiddleFinger_03". Now repeat the rotate and stretch operations again to place the new bones correctly, as shown in Figure 5.

Now set up the rest of the fingers. Repeat the clone, rotate and stretch operations for the RingFinger, PinkyFinger, and ThumbBone. Your final bone layout should look like Figure 6. To make sure you have the correct hierarchy and naming, compare your results to Figure 7.

The last step in building the skeleton is to lock out movement and rotation for the bones. Click on the Animate Tab. Click on the Keys sub-tab. Next to the X, Y, Z, H, P, and B there are colored boxes. To hide the rotation and movement handle for the bones, just select the bone you want to edit, then right-click in the colored box for the channel you want to turn off. First, turn off the Move handles for all the bones. Next, turn off all the handles for the bones with names ending in "_MAIN". Leave Heading and Pitch on for the base of the fingers, "_01". For all the other bones, leave just the Pitch on. Now everything should be OK. Scroll through the bones to make sure you have the correct handles hidden.

Adding Expressions

The next section of this tutorial shows how to add expressions for animating the hand. You will add Sliders to control individual finger rotations, rotations of the whole hand, plus one Slider to switch from Slider control to standard Forward Kinematics, just in case you need to manually position the fingers. This setup gives you the fastest means of posing the majority of hand motions while retaining the freedom to fine-tune unique hand poses when necessary.

First, you need to add a slider object to the scene. Go to the File Tab and click on the Add Slider button. Rename it by selecting the Name sub-tab. In the name field, type "Right_Hand_Slider" and press Enter.

Next, name the sliders for the fingers. Click on the Animate Tab, then make sure the Keys sub-tab is selected. You need seven sliders, but by default only six are on. Scroll down until you see the seventh button. Right-click in the gray field with the check mark, as indicated by the red arrow in Figure 8. That will enable the seventh slider for you to use.

Rename the sliders so you know what they are. To the right of the check boxes are the names: xpos, ypos, zpos, heading, etc. Just highlight each name field in turn, and change the name to something useful. I start with the thumb, then work my way around. Highlight "xpos" and change its name to "ThumbR". Then change "ypos" to "PointerR". Keep changing the names down the list like this: MiddleR, RingR, PinkyR, ALL_R, and finally "Sliders<-->FK_R". Your results should look like Figure 9.

I always parent things to keep them organized, so I parented the slider to the hand that it is going to control. Just left-click on the Right_Hand_Slider object and hold down the mouse button, then drag the object up over the hand object and release the mouse button.

Now you need to set up the expressions for the sliders. Select the Command Tab. When I am working with expressions in P:M I like to be able to see as much of the expression as possible. To do this, just press the F3 key to expand the left column of the interface to fill up more of the screen.

First you need to set up the slider values, then limit the range of motion for the fingers. While in the Command Tab, make sure you have the Expression sub-tab selected, and then click on the New Expression button. The result will be a new expression called "Exp". You should give descriptive names to all your expressions so you don't get confused about which is which. In the Name field, select "Exp", change the name to "Rthumb" and press Enter.

Select the Functions sub-tab. The Thumb expression is a little bit different from the rest of the fingers because it doesn't have to bend as far, so we will do it first.

From the Item pulldown at the bottom right of the P:M interface, select the slider object "Right_Hand_Slider". Click on the "ThumbR" slider. Now left-click on the button at the end of the Text field. It should read "[Right_Hand_Slider:xpos]". If that is what it reads, then right-click the same button. This will add that text to the Exp field. Click on the "ALL_R" slider. Check the text again by left-clicking on the button next to the Text field. Right-click the button if the text is correct, and the text will be added to the Exp field.

You need to tell P:M what to do with the two sliders. You want to add the two values together, so you can position all the fingers with the ALL_R slider, then add or subtract motion with the individual finger sliders. To add the slider values together, click in the Exp field and add a "+" between the two. Your expression should look like this:

[Right_Hand_Slider:xpos]+[Right_Hand_Slider:bank]

Repeat these steps to add expressions to the other fingers, but add a multiplication factor of 1.5, like this: (next finger slider + ALL_R slider)*1.5. For example, the RPointer expression should look like this:

([Right_Hand_Slider:ypos]+[Right_Hand_Slider:bank])*1.5.

Note the parentheses; they tell the expression to add the two slider values before it multiplies by 1.5. Otherwise, it would multiply first, then add. And you thought you would never use that stuff they taught you in math class! How did I come up with the 1.5 for the rotation value? By that old standby, trial and error. I moved one of the finger sliders all the way to the right, and the finger didn't rotate all the way shut. I guessed a value of 2, and that was too much. I then backed off the multiplier

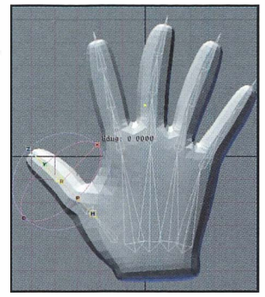
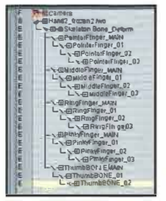


Figure 6: Completed hand skeleton.



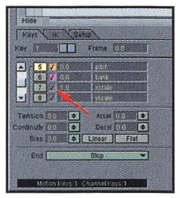


Figure 7, above left: Naming and hierarchy for the hand skeleton shown in Figure 6.

Figure 8, above right: The seventh slider to be enabled in the Animate|Keys sub-tab.

Figure 9: Renamed sliders.

Figure 10: Expressions for all five finger sliders.

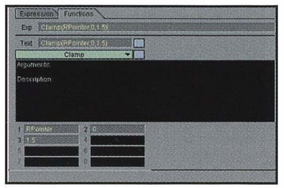


Figure 11: Values entered into the three enabled Clamp function fields.

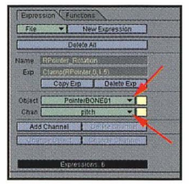


Figure 12: Expression sub-tab showing PointerBONE01 and Pitch selected.

Figure 13: Final expression for RPointer.

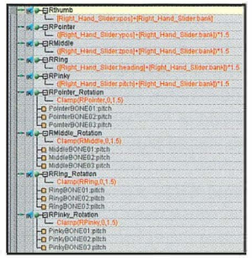


Figure 14: Completed finger expressions.

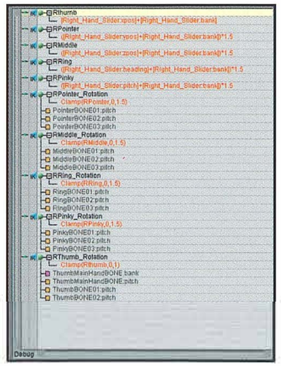


Figure 15: Slider and rotation expressions for the entire hand, including the thumb.

value until I reached the desired rotational value to close the finger correctly.

You should be able to complete the rest of the finger expressions on your own. Your results should look something like Figure 10.

Now you need to apply those slider values to the finger bones. You also need to limit the range of motion so the fingers can't bend backwards or too far forward. You can do this with the Clamp expression. Clamp will return one of three values (float, min, max). This is where things get a little complicated. The first value will be the value we get from our sliders. The next two values would be the minimum and maximum rotation values. We are going to start with the Rpointer finger.

Go to the Command Tab. Select the Expression sub-tab. Click the New Expression button. In the Name field, type Rpointer_Rotation. Select the Functions sub-tab. In the function pulldown, select Clamp. When you select clamp it enables three fields under the pulldown. Enter these values into the enabled fields: 1. Rpointer, 2. 0, 3. 1.5, as shown in Figure 11.

Left-click on the button next to the function Clamp. Your expression should look like this: Clamp(RPointer,0,1.5). If it does, then right-click on the same button to make that your expression.

Now you need to tell the expression what bones and channels you want it to affect. Click on the Expression sub-tab. Select the expression "Rpointer_Rotation". In the Object pulldown, select the bone and channel you want to affect. In this case, you want to select the PointerBONE01 and the channel pitch, as the red arrows indicate in Figure 12.

Click the Add Channel button. Repeat these steps for PointerBONE02 and PointerBONE03, also affecting the pitch channel. Your final expression should look like Figure 13.

Now set up the Clamp expression and bone and channel selection for the other fingers, but not the thumb yet. Your finger expressions should look like Figure 14.

For the thumb, you need to set up the same expression with just a couple of modifications. You want the max rotation to be 1, not 1.5. You want the expression to rotate the "ThumbMainHandBONE" on the pitch AND the bank. This gives us a nice natural rotation for the thumb. Add the pitch channel for "ThumbBONE01" and "ThumbBONE02". Your final expressions should look like Figure 15 for the sliders and rotations.

The final step is to add the expressions to enable you to switch from slider animation to standard Forward Kinematics. This uses the KeyFader function, which enables you to fade from procedural animation to keyframed animation. This is the most complicated part of the setup, so pay attention.

Make a new expression with the name "Rthumb_S<-->FK". Switch to the Functions sub-tab and select KeyFader from the pulldown menu. In field one, you want to choose ThumbMainHandBONE to be the object/bone to be affected. In field two you want the starting point, which would be "IN" for the Start of the animation. In field three you want it to read "OUT" for the finish or OUT point of the animation. In field four you want to select which channel for the KeyFader to fade. Pull down the menu and select "motion" for ALL the motion of the object. The fifth and last field is to tell the expression what percentage/object will control the fade. You want the last slider to control the fade, so it should be [Right_Hand_Slider:xscale].

With all that set up, right-click on the button next to KeyFader to set that expression into the Exp field. Wait, you aren't quite done yet! That expression is only going to work for the ThumbMainHandBONE. We need to add the other bones in the thumb.

Select the next bone in the chain, ThumbBONE01, from the function 1 slot. Right-click on the button next to the KeyFader function again. One last bone! Select "ThumbBONE02" from the pulldown menu and right-click the button next to the KeyFader function.

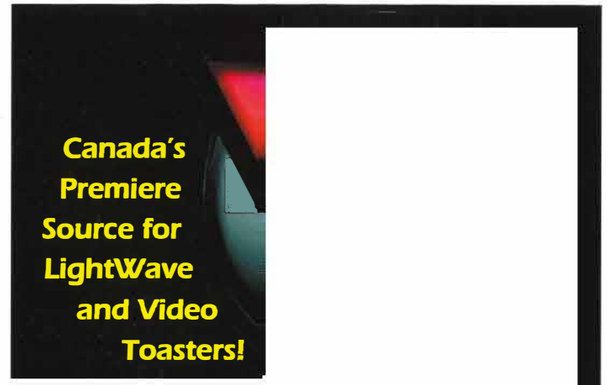
Now you have this unbelievably long expression. In order for all the parts of the expression to work together, you need to add a "+" between each of them. Go into the Exp field. Scroll over until you get to the spot where two expressions meet. You should find a space between them. You want to replace that space with a "+". Here is what your final expression should look like:

KeyFader(ThumbMainHandBONE,IN,OUT,motion,[Right_Hand_Slider:xscale])+KeyFader(ThumbBONE01,IN,OUT,motion,[Right_Hand_Slider:xscale])+KeyFader(ThumbBONE02,IN,OUT,motion,[Right_Hand_Slider:xscale])

Now you need to set up this expression and slider for the other fingers. Start with PointerBONE01. You don't need the MAIN bone because it isn't part of the sliders' animation. You'll notice when you go back to the Functions the KeyFader is still selected with all of your options still in the fields. All you have to do is change the object to the different finger bone. Once you are done with those expressions, that's it! You should now have a complete hand setup that looks great and is simple and efficient to animate.

I would like to thank Ron Griswold from Station X Studios for helping me to revise my expressions and showing me the light with KeyFader. Big thanks also go to Station X Studios for releasing Project:Messiah to the public!

John Riggs is Technical Director at Vision Scape Imaging. You can email him at jriggs@vision-scape.com or visit their website, www.vision-scape.com.



JIMMY NEEDS YOUR HELP!!!
...FOR A NEW 3D ANIMATED FEATURE FILM FROM NICKELODEON IN ASSOCIATION WITH O ENTERTAINMENT FOR PRODUCTION IN SPRING 2000!
DNA PRODUCTIONS, INC. IS LOOKING FOR...
- 3D CHARACTER ANIMATORS
- 3D MODELERS
- LIGHTING GURUS
- SUPERVISORS
- ARTISTS
BOY GENIUS
REFER TO OUR WEBSITE FOR MORE INFO ON POSITIONS & SUBMISSIONS: WWW.DNAHELIX.COM
DNA PRODUCTIONS, INC.
2201 W. ROYAL LANE, SUITE 275
IRVING, TEXAS 75063
COPYRIGHT 2000 NICKELODEON

Compositing LightWave Objects

by Phil South

As a 3D animator you are doubtless aware that a good proportion of modern special effects are done using 3D programs. If you aren't aware of that then I suggest you go right back to issue one this instant and bone up on the entire history of 3D CGI special effects...

You back? Okay, while you were gone I devised a simple tutorial to introduce you to the concepts of compositing 3D objects with the real world using the sharpest chisel in the box, LightWave.

The Basics

For a start this principle works for both 3D objects in real world scenes and 3D scenes containing real world objects. Obviously the latter is a LOT more complex and involves green screens and Video Toaster NT, and we don't want to get into that quite yet. (I have just received my VTNT at the time of going to press and I must say it's starting to make me think that green screen work could make a viable hobby now that the rocketry season is at a close...) But I digress.

What we are going to do is take a couple of background plates and add a 3D element to them both. In this instance I have used a couple of copyright free PhotoCD images, but you would normally either take them yourself on the set or be supplied with them from the makers of the film or video. The two shots I've chosen are a little tricky because one is quite dark and backlit and the other is sunny and lit from the side.

The techniques I've used in these shots are very similar, even though the pictures are lit very differently. The techniques are however just starting points for experimentation, because more than any other 3D technique, this is very much a "season to taste" exercise. You have to check it with those two glossy marbles you have stuck into the front of your head and see if it looks right, and if it doesn't: change something.

These two shots are prototypes which I've been working on for a movie I'm making, and the cheat, if that is the right word, for "selling" these shots is the application of a third party filter called Aged Film from DigiEffects (www.digieffects.com). The reason for this is twofold.

First adding any kind of grain filter, even relatively mild film grain, to a composite blends the two elements together in a really nice way, and this is a common feature of most compositing jobs, particularly for film work. Of course digital composites are always pretty clean, and of course the 3D elements are cleaner still. Adding film grain makes both the 3D elements and the background plate look very much "of a piece" as we say in the UK, or looking like they were made at the same time on the same piece of film.

The second reason for adding grain in this case is to make the composites look like grainy 8mm film shots of UFOs. Another option would be to add video noise and lines (Videolook from DigiEffects is another cool option) to give the look of amateur video. Adding camera shake to a background plate which is of course a little larger than the screen area is also useful, especially if the shot is supposed to be hand held.

Shot One: Up and Down at Sunset

The first shot is a sunset shot over a field in what looks to my eyes to be the UK somewhere. The light is coming from the sun and the shadows are a sort of dark brown color.

Our first job is to estimate the color of fog we are going to use. Why fog? Well fog gives us the ability to blend a 3D element into a scene with the right color of haze, which in real life is caused by aerial particles. This effect is called "aerial perspective", and the addition of it gives any 3D shot the look and feel of an outdoor scene by mimicking particles in the air. Obviously, as you get closer to the horizon the color of the haze needs to match that of the background plate. It would be a good idea to always begin by loading the plate into a graphics program like Photoshop and taking a sample of the shadow color at the horizon, in the middle and in the foreground to give us a starting point for our haze colors.

In this case our horizon color is: R=116, G=94, B=82. In the middle it's R=71, G=57, B=46 and in the front it's R=22, G=18, B=15.

Let's load in our background plate into LightWave and get started. Position the UFO, in this case the Adamski model from the LightWave default object set, and have it drift gradually downwards from a point nearly level with the sun to a point near the ground over about 125 frames. You can introduce a little wobble to it, which looks authentically like an anti-grav drive even though such things would be a lot steadier. You could even have it sail to the ground like a leaf, but I don't mind about that stuff right now. Making the animation look real is up to you; I'm more concerned about how tacked-on this looks.

Speaking of which, try a render at this point. Hmm. Looks pretty darn fake, yes? And so it should. The light isn't coming from the right direction and the ambient lighting is up too high. Take the ambient lighting down to our regular level of 5%. You could even add another light coming up from the ground with a tint taken from the RGB value of the ground in the plate.

Our first job is to add some fog, and as this UFO is only moving up and down not back and forth, the fog can be the same value all over. Open the Effects Panel and add a Linear Fog with a minimum distance of 0m and a max of 500m. That covers everything. Then make the Fog Amounts both 50%. In the fog color field enter our middle fog color of 71, 57, 46. Now do another render. Ah, now that looks better, but a little dark perhaps. Go back to the Fog Color setting and shift the color up a little. Just grab the little black arrow at the side of the color picker and make the thing a lot lighter. I got to 146, 118, 95 before I was satisfied.

A little added finessing: I put the light where the sun is in the picture (it helps if you have the background image turned on in Layout) and added lens flare, but turned off everything except Lens Reflections. This way you get the lens reflections going in front of the UFO when it goes past them. Add the Aged Film filter and voila! Instant class. I pulled the sliders in Aged Film around a bit, having no scratches or dust and reducing the grain a little to make it more tolerable.

Shot Two: Back and Forth up the Avenue

Now then that was good, but what if the fog values are different? What if your shot involves an object going from near to far (or vice versa) and you need to blend the object differently as it travels through the scene? You've probably guessed that of course the fog values have to be different at the front of the scene (negative Z) than they are at the back (positive Z).

Take our next example. We have a lovely picture of an avenue of trees. What we are going to do is take another UFO (the Klaatu one this time) and sail it up the avenue from the distance to the foreground. Again the shading on the UFO is not so good so I adjusted it a little. Making it mirrored with a reflection map of the trees was my first thought, but in the end I settled for my basic metal. (See issue 15 for details about that.)

At the back of the scene the shadow value is a nice dark green, something in the order of 80,122,102. So let's make our fog value that figure when the saucer is at the back of the screen. Even though the scene is quite clear we should still use a little fog in the middle and at the front, so set the minimum distance value to the mid point and have that at 2%. Even on a clear day there is some matter in the air.

My UFO started out 80m away from the camera, so that was my maximum fog distance, and I set the max value to 45%. The minimum distance was about halfway so that was 40m and our minimum fog value of 2%. Render the object at various distances and check that the fog and the lighting are good, and edit the values to better suit the effect at certain points in the animation.

And finally don't forget to light your object properly. Remove all but 5% of the ambient light, and put in lights for the sun, the sky and the ground to mimic the way light plays around an object in a daylight scene. Don't forget to ramp the lights up and down when the UFO moves into shadow, or better yet put front projected tree trunks into the frame to cast proper shadows on the UFO as it travels along the avenue.

Use Your Eyes

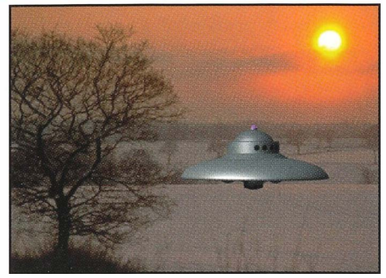
Compositing is a real minefield, but it's not really that hard to learn. As I said at the start, the problem is that although there are certain techniques you can use time after time, these are usually merely starting points. As with almost any other 3D (or for that matter any graphics) work the important thing is to use your eyes. You must look at what is there and replicate it, like a painter looking at a still life. You have to copy the lighting and more specifically the "mood" of the lighting in the picture and apply it to your objects.

Mood is an intangible, but practice will provide you with the tools you need to make it happen. There is no plug-in for experience. You can't use the same effects on each job and expect it to work, but you can what I call follow similar lines of inquiry. Use fog by all means, but what if there isn't any haze in the picture? What then? Well, a little fog (and even the soft filter in the camera panel) can help to blend your objects in. Added grain also helps to bring it all together, but once again, what if you are working in 70mm and there is no grain to speak of? These are challenges you'll have to face, using all the tools in your palette.

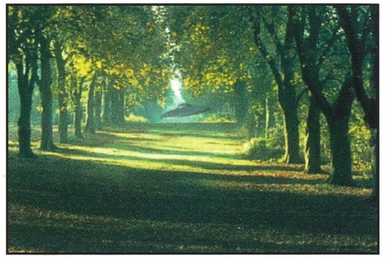
The answer? Try everything. Use the Soft Filter, or even use Photoshop filters (using the Photoshop plug-in of course). Process the 3D objects and backgrounds separately and process them differently to arrive at the same result. Use your Alpha channels. And also don't underestimate the power of a foreground object behind which your 3D elements can pass to sell the idea of them being IN the shot rather than on top of it. Pull a matte on some tree branches or a church spire and pass things behind them.

Compositing is all about blending, blending the join between your 3D objects and the scene into which you plop them, but also blending the more important line between fantasy and reality.

Phil South is a freelance animator and web designer based in the West Country of England. He can be reached via his web site on: www.ii3d.freeserve.co.uk.

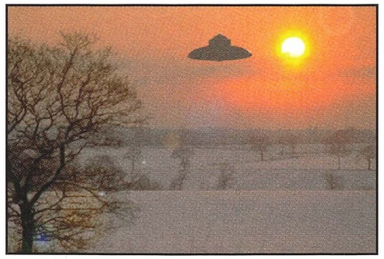


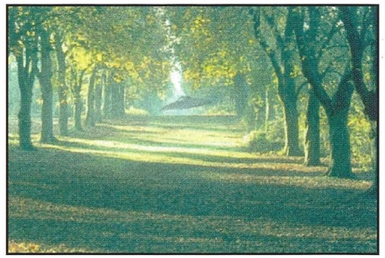
The original renders show how tacked-on the objects look before you start to blend them in. Fog and lighting are crucial tools.



At the top of its flight, the UFO is close to the sun, and there is little light playing around the sides. Then, at the bottom, the ground light starts to illuminate the rim and the sun starts to show over the top.

At the midpoint the fog is at its minimum. Although the air is pretty clear there are still some airborne motes. Close to, the fog is not really there at all so you have to make sure your lighting is really close to the original plate.





The film filter, with less brightness and contrast, makes the composite complete.

With the application of Aged Film, the shots look a lot more realistic, as the 3D elements and backgrounds receive both grain and a little adjustment of contrast.

Take One

by Lee Stranahan

In this issue, I want to give you ten tips to keep your LightWave projects organized. These aren't animation tips, but they are important tips to getting more done with LightWave. Most of them are more helpful when working with a group of animators, but I hope you'll find a way of thinking about your LightWave projects that will work in whatever size studio you have.

The brave new world of digital media is a very egalitarian place. Sure, a lot of companies say they want people with degrees. But the fact is, if you have a great demo reel, and you get it into the hands of the right people, you'll be able to get hired. That's as it should be, really, because a lot of LightWave animators didn't learn their trade at any school; they learned by working night after night in their bedrooms.

If you're still a "bedroom animator," you've probably never had to deal with the issues of a complex, multi-shot project. On this kind of project, there are several levels of approval that each shot needs to pass through, and you're dealing with a number of different animators, model makers, texture artists, and compositors. I'm not saying that the real world is harder than bedroom animation - in some ways, it's easier - but it is different. After you get your animation and modeling chops down, and you become comfortable with working methods in LightWave, it's a good idea to broaden your horizons. Start doing some reading and research about business issues like organization and the economics of 3D graphics.

Use a Different Filing System

NewTek uses a file structure that puts all objects in one file folder, all scenes in another, all textures in another, and so on. If that system works for you, great. The problem with this filing system is that it makes it difficult to gather all the elements for a show in one place. Additionally, you end up with a ton of extraneous files; objects and images that bear no relationship to any project you're working on. That's why a lot of people use a system where they create a master folder for the project, and then create subfolders for objects, images, scene files and the like. This way, all the appropriate files are in one place, with no extra junk.

Use The Script's Scene Numbers...Or Don't.

Every production script has scene numbers; they are usually on both the left and right margins. These scene numbers are used by the production team to plan the shoot. The problem with a script's scene numbers is that they often aren't very useful for you as animator. Let's take a look at a fake script page as an example.

We have three very short scenes. Let's assume all the Giant Robots are LightWave elements, and that the Enemy Fortress is a real location, not a CGI element. As a little exercise, try to do a visual effects breakdown - how many visual effects shots are there in scene 34, 35, and 36?

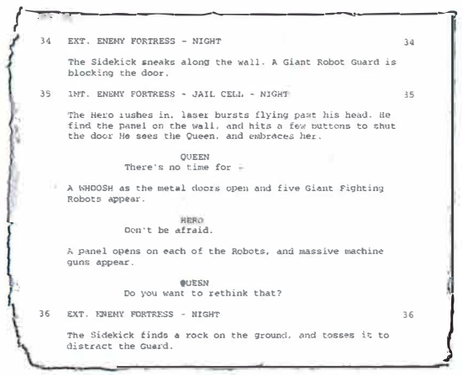
And the answer is...(drum roll)...I don't know.

Sorry, but there's no way I can know without discussing the script with the Director, Producer, or other decision-maker. There's no way you can, either. Bummer, but it's true.

As an example, Scene 34 is extremely short - just two sentences, and the Giant Robot is only mentioned in one of them. So how many visual effects shots are there in this scene? Again, I don't know.

Let's look at a few options.

- Maybe the director wants to cover that scene with one shot. This would be one visual effects shot.
- Maybe there are two shots - one of the Sidekick sneaking, and then his POV of the Giant Robot. That's two shots, one of which has visual effects.
- The director could cover Scene 34 with five shots. First, an establishing shot of the Fortress, with the Robot visible. Next, a medium shot of the Sidekick near the wall. Then, the Sidekick's POV of the Robot. After that, a close-up of the Sidekick looking concerned, then looking back towards the Robot. Finally, a closer shot of the Robot. That's five total shots, three or four of which seem like they'd have visual effects.



34 EXT. ENEMY FORTRESS - NIGHT 34

The Sidekick sneaks along the wall. A Giant Robot Guard is blocking the door.

35 INT. ENEMY FORTRESS - JAIL CELL - NIGHT 35

The Hero rushes in, laser bursts flying past his head. He find the panel on the wall, and hits a few buttons to shut the door. He sees the Queen, and embraces her.

QUEEN
There's no time for -

A WHOOSH as the metal doors open and five Giant Fighting Robots appear.

HERO
Don't be afraid.

A panel opens on each of the Robots, and massive machine guns appear.

QUEEN
Do you want to rethink that?

36 EXT. ENEMY FORTRESS - NIGHT 36

The Sidekick finds a rock on the ground, and tosses it to distract the Guard.

Figure 1: Example script page.

There you go - three different ways to cover two little sentences, and there are probably half a dozen more variations that might make sense. Our organizational problem is, how do we identify these different visual effects shots within one scene? There are two schools of thought. One says, use the scene number and then a letter or number suffix. So, we might have 34A, 34B, and 34C. The other school says, use anything you want. If scene 34 contained the first three visual effects shot in the show, we might call them FX1, FX2, and FX3. Personally, I prefer the scene number scheme based on the script's scene number, because it cuts out an extra set of numbers.

Take A Trip To The Office Supply Store

You know you've hit a career milestone when you start visiting the office supply store more than you visit the computer store. There is something satisfying about seeing row after row of devices designed to make your life more organized. So, buy some!

I like to make a notebook for each project I work on. I start with a nice little binder. What goes in it? Things like...

- Phone numbers and email addresses of all the key people on the project, including the client and all the animators.
- Copies of receipts for any equipment purchased for the project. (Hey! We need a new hard drive!)
- A complete copy of the script, or at least all relevant script pages.
- Shot sheets (see below)
- Master schedule (see below)

If you have a color printer, you can also have fun making a quick little cover sheet for your notebooks. And given the access you have to high end graphics tools, there'd better be a cool picture!

Give Your Shots Their Own Page

One school of thought says that every single shot should have its own sheet in order to keep track. I know, paper is so 20th century. But there are still advantages to good old processed wood chips and a pen or pencil. (For one thing, paper doesn't crash.) You may have to create a spreadsheet, or use some other digital method to keep track of a project, but you'll also want pieces of paper. Plus, you just bought a cool binder, so you need to fill it.

The shot sheet should have a brief description of the shot, the storyboard sketch(es), the page number where the scene begins in the script, space to put dates of revisions (and notes about each one), and signature lines or initial boxes to indicate the various stages of approval.

Chart Your Progress Through The Project

You'll also want to keep track of the whole project, shot by shot. As each shot is approved as to model, texturing, lighting, animation, final render, compositing, and so on, it'll be checked off on the master sheet. The master also keeps track of which animator is working on which shot.

Keep File Names Short and Sweet

Long file names can cause all sorts of problems, even though the eight-dot-three naming convention of DOS (no more than eight characters, a dot, and then a three-letter suffix, such as FILENAME.TXT) is supposed to be long past us, because it's not totally past us. There are still cases when your long file names will get truncated and that causes all sorts of problems. One common example is when you transfer files from a Windows machine to a Macintosh; the DOS-to-Mac converter often forces the file names to 8.3. Shorter file names are also easier to read in a file requestor.

Use LightWave's Data Overlay Function

Once you start rendering animations, you are suddenly creating a potential organizational nightmare for yourself. Let's just assume we're doing a seven second shot, and let's further assume that we're working at NTSC video resolution: 30 frame per second. So, each time we render out a test animation, we are rendering 210 frames, and if we did four tests of the animation plus a final render, we've got over a thousand new little bundles of joy to deal with. A well-thought-out filing system and clear file naming conventions will help you keep all these files organized.

There is also an issue when you view the animations, namely, telling which animation you're watching. LightWave has a nifty little feature in the Render panel called Data Overlay. Generally, it's clearest to just enter the scene name, so you'll always know what scene you're looking at. Don't use Data Overlay on final renders; you knew that, right?

Give Disposable Files Unique Names

Sometimes you have to create files that you know you won't ever use again. For example, a motion file that you save just to copy another object's motion. Get in the habit of giving all these files the same name prefix. I use "KillMe" so If I'm in cleanup mode and I come across KillMe1.mot, I know that it's safe to erase that file.

Clean Up Your Mess

It's a good idea to schedule clean up time; otherwise, the mass of files will grow and eventually eat you. You need to make time, it generally won't just appear. Make an appointment with yourself, or get in the habit of cleaning your files up before you log off. Again, it's whatever works for you. Create a system that works, and use it.

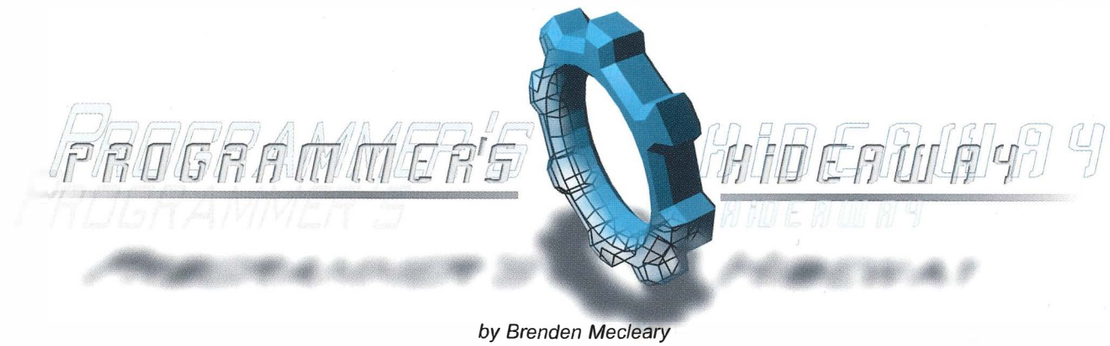
Save Lots And Backup

Models and scene files generally don't take up too much disk space, so it can't hurt to save multiple revisions. And I don't really need to tell you to make frequent backups, do I? CDs are cheap, dude, and there are so many backup options available that's there no excuse for not doing it.

There you go: some tips to make your LightWave life a little easier. See you in the paperclip aisle!

Lee Stranahan (www.stranahan.com) is a filmmaker, photographer, visual effects artist and teacher who has taught thousands of people how to get more out of LightWave through seminars, videos and CDs. He lives in Pasadena, CA.

LightWave Envelope Support for Plug-ins



by Brenden Mecleary

Howdy, everyone. Welcome back to the hideaway. Grab a cup of coffee and step into my cubicle. No, not that close. OK, while everyone's breathlessly awaiting LightWave 6.0, we still have plenty to keep us busy in our trusty old LW 5.6.

Today, we're going to look at animating your plug-in's parameters via the envelopes global plug-in. But before I get into that, I'd like to answer a question that I've seen asked several times by people new to LightWave plug-in development. This question is, "How do I make plug-ins in C++?" A close runner-up would have to be, "How do I make plug-ins in MFC?" but I generally frown upon that sort of thing. (If you follow this column at all, you probably know how I disapprove of tying yourself to one platform.)

The basic problem with writing your plug-ins in C++ is that LightWave is written in C (or at least, its SDK is). In order to get certain things to work, a C++ compiler will "mangle" the names of functions and variables. This allows things such as function overloading, and also makes it possible to enforce link-time type checking. Normally, this happens transparently to the programmer. The trouble is, a C compiler can't recognize these "mangled" names, and thus is not directly compatible. However, by using 'extern "C"', you can force the C++ compiler to generate object code that is compatible with C.

So what do you have to do to your code to make it work? Fortunately, using C++ in your plug-ins is very easy. The only thing you have to do differently when writing your plug-ins in C++ is to give your server descriptor C linkage, using 'extern "C"'. In C, your server record would look like this:

```
ServerRecord ServerDesc[] =
{
    {"ShaderHandler", "MyShader", MyShaderActivate},
    {"ShaderInterface", "MyShader", MyShaderInterface},
    ...
    {NULL}
};
```

Whereas in C++, it should look like this:

```
extern "C"
{
    ServerRecord ServerDesc[] =
    {
        {"ShaderHandler", "MyShader", MyShaderActivate},
        {"ShaderInterface", "MyShader", MyShaderInterface},
        ...
        {NULL}
    };
}
```

Likewise, if you're using the "quick" server descriptor method, it changes from this:

```
extern char ServerClass[] = "ShaderHandler";
extern char ServerName[] = "MyShader";
int Activate(...);
```

To this:

```
extern "C"
{
    extern char ServerClass[] = "ShaderHandler";
    extern char ServerName[] = "MyShader";
    int Activate(...);
}
```

To tell the truth, I don't really like that method of defining plug-ins very much. It forces you to have only one plug-in per file, and the added convenience is negligible.

Finally, if you are determined to write your entire plug-in yourself (and I mean every last line of code), you will have to specify C linkage for the _mod_descrip structure. But there's no real advantage to doing this, and I don't recommend it.

Envelopes Global Service

For a long time, plug-in developers were stuck when it came to making any part of their plug-ins variable over time. LightWave proper had envelopes as a convenient way to animate parameters, but there was no such service directly available to plug-ins. This led to several work-arounds, both creative and not so creative.

Quite possibly the worst solution was the "rate of change" value - e.g. parameter X starts at value Y, then increases by Z units per second. Think of LightWave's "texture velocity" setting. Then think of trying to figure out where texture pixel (25, 183) will be in 74 frames. Now you understand why this method is so awful. (And why NewTek added the option to attach textures to null objects.) About the ONLY good thing that can be said about it is that it's easy to implement. Otherwise, it's non-intuitive, requires much more mental work, and even programmers hate using it.

The next solution, one that only saw light in a few plug-ins, is a sort of keyframe system. The plug-in interface lets the user set keyframes, and navigate among them (usually by means of "next" and "previous" buttons.) The LumeTools package used this method for a while, but the envelope global came out during development, and it was decided that the plug-ins should be re-written to use it instead. I can't say I'm terribly fond of the keyframe method either, but it does give the user more control of how things progress during the course of an animation.

Just before the envelope control was released, developers had more or less settled on using null objects to animate values. This was actually a relatively tolerable solution. Perhaps the nicest thing about it is that users don't have to go back into the plug-in panel to change the settings. In fact, the interruption in workflow to change a setting is negligible. It also allows users to use LightWave's own envelope editor (in the form of the motion graph) for precision control. By using item motion plug-ins, users can also generate very complex inputs for a plug-in's settings. And a good implementation could allow up to six different values to be attached to a single null object. Using null objects to animate values is still an idea that has merit; in fact, some developers are giving users a choice between using a null or an envelope when animating a setting.

So, enough with the history lesson. The envelopes plug-in is now available as part of the LightWave SDK. To get an envelopes interface, #include LWEnvlp.h in your source, then call the LightWave global function and ask for ENVELOPE_ACCESS_NAME. Usually, you'll need to acquire it and release it separately, since you'll need the envelope services across the context of several functions. Let's take a look at what is returned from the global function:

```
ypedef struct st_EnvelopeHandler
{
    int api_version;
    Envelope *(*create)(LWError *);
    void (*destroy)(Envelope *);
    LWError     (*copy)(Envelope *from,Envelope *to);
    LWError     (*load)(Envelope *env,char *filename);
    LWError     (*save)(Envelope *env, char *filename);

    void (*evaluate)(Envelope *, LWTime, ChanVec);

    //Return 1 on OK, 0 on cancel(undo), -1 on remove
    //client must destroy() Envelope and copy
    int   (*Interface)(Envelope *, ehInterfaceData *dat);

    // API V3
    LWError     (*LWLoad)(Envelope *env, const LWLoadState
*);
    LWError     (*LWSave)(Envelope *env, const LWSaveState
*);

    //Allocate and fill key
    Keyframe *(*createKey)(int step,ChanVec cv,int chans);

    //Insert key in envelope, replacing any existing key at the
same step
    void (*putKey)(Envelope *env,Keyframe *key);

    //Return key in envelope at step, or NULL is no key there
    Keyframe *(*getKey)(Envelope *env, int step);

    //Remove key from envelope and free
    void (*destroyKey)(Keyframe *key);

    // API V4
    //Load env from open stream
    LWError (*loadStream)(Envelope *env, FILE *fp);

    //Save env from open stream
    LWError (*saveStream)(Envelope *env, FILE *fp);
} EnvelopeHandler;
```

The EnvelopeHandler structure contains pointers to all the typical functions - create, destroy, copy, load, save. There are also some functions for getting specific keys and performing operations on them, should you ever need to do so. And, of course, the really important functions evaluate() and Interface(). One of the nice things about the envelope services is that they map almost directly to C++ class operations - construction, destruction, copying, etc. Other operations fit in nicely as member functions.

The first thing to do, before doing anything with an envelope is, of course, create it. This returns a pointer to the envelope's data. There's nothing to prevent you from digging around inside these structures, but there's rarely much need to do so. Once you have the envelope pointer, you can pass it to any of the other handler functions.

Generally, using the envelope services is very straightforward. Create your envelope when your plug-in is created, destroy it when your plug-in is destroyed, load it when your plug-in is loaded, and (can you guess what's next?) save it when your plug-in is saved. The only slightly tricky part comes when it's time to handle the interface.

Let's look at setting up the structures for creating an envelope interface. Here are the relevant structures from LWEnvlp.h:

```
typedef struct st_channelDescription
{
    char *chname;                   // Label for popup
    float min, max;                 // acceptable value ranges
} channelDescription;

typedef struct st_ehInterfaceData
{
    LWPanelFuncs                *pfunc;
    const char                  *title;
    channelDescription      chDesc[MAX_CHANNELS];
} ehInterfaceData;
```

When you call the Interface() function provided by the envelope services, you'll need to hand it an ehInterfaceData pointer. The first two fields in this structure are pretty obvious: a pointer to a LWPanelFuncs service, and the caption of the envelope editor window. The final field is an array of channel descriptors. Each channel descriptor has a title, and a minimum/maximum value. To indicate that there are no more channels in the envelope, add an extra channel with a NULL channel name pointer.

This is an example of how to set up an envelope interface data structure:

```
ehInterfaceData envInterfaceData =
{
    pPanelServices,
    "My envelope",
    {
        {"Red", 0.0f, 255.0f},
        {"Green", 0.0f, 255.0f},
        {"Blue", 0.0f, 255.0f},
        {"Alpha", 0.0f, 255.0f},
        {NULL}
    }
};
```

Now, when driving the interface for your envelope, the first

thing to be aware of is that you must make a copy of your envelope, and let the user edit the copy. This is because all changes the user makes are actually applied to the envelope while it's being edited, even if they cancel. So, make a copy, let the user edit that, then only apply the changes if the user clicks OK.

The next thing to be aware of is how you display the button for the envelope. This is actually somewhat trivial, but the default behavior of a BOOLBUTTON_CTL (the most common control used to display envelopes) is to turn on with the first click, off with the second. However, envelopes don't want that behavior. Envelopes want the button to be on when the envelope is on, off when the envelope is off, and to always show the envelope editor when clicked. If the envelope is already on and the user clicks OK after editing it, the button should stay on. This is very easy to solve: just be sure to always set the state of the button explicitly. However, it's easy to overlook when writing your envelope code, so don't forget.

One last little gotcha when using the interface code: the handling of channel names. Setting up the interface handler is a bit ugly, but not too bad. However, sometimes you'd like to isolate it in its own function, like so:

```
//Function to set the titles of envelope channels.
//BAD - DON'T DO IT THIS WAY!
void SetEnvInterface(ehInterfaceData * pInterfaceData)
{
    pInterfaceData->title = "My envelope";
    pInterfaceData->chDesc[0].chname = "Channel 1";
    pInterfaceData->chDesc[1].chname = "Channel 2";
    pInterfaceData->chDesc[2].chname = "Channel 3";
    pInterfaceData->chDesc[3].chname = NULL;

    //...other setup for the envelope's interface.

    return;
}
```

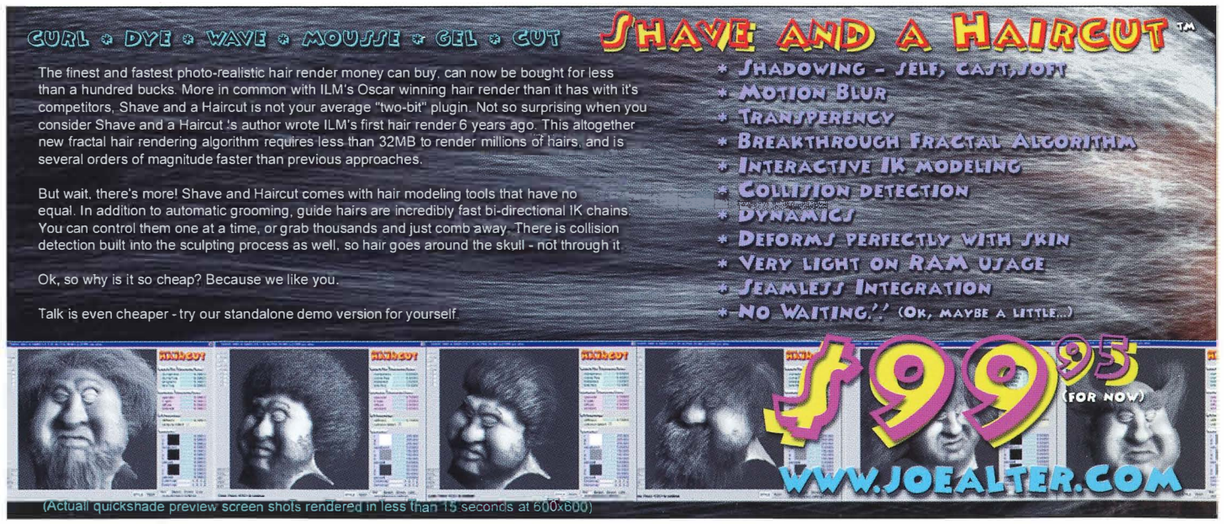
If you do something like this, you'll be in for a nasty surprise when your plug-in crashes and burns, taking LightWave down with it. The reason is that, like many of the LWPanels services, all those title pointers in the interface data structure are local pointers. When the SetEnvInterface() function listed above returns, all the pointers to those strings become invalid. When the envelope service tries to access them, it will cause an access violation. My solution to this problem was to keep some buffers for those strings in my envelope wrapper class. The interface titles then point to those buffers. In this way, the titles can be set at any time. The overhead is negligible, about 8.5 K per object.

I've included the source code for an envelope wrapper on my site at http://www.gsidigital.com/bman/binaries/keyframe_16_files.zip. It may not be completely polished yet - I only ran some cursory tests on it, which are also included - but it should serve as a good starting point. Listing 1 is the header that defines the client interface for the wrapper. Listing 2 is the source code, a bit long, and less thoroughly commented than I'd like. Listing 3 is a tiny plug-in that tests the envelope wrapper class.

The big requirement to create an instance of the envelope wrapper is a pointer to the LightWave global services function. Usually, you provide this to the constructor so that everything can be initialized right away. Some times this is not possible; for example, when creating arrays of envelopes. For this reason, it is possible to create an envelope wrapper instance without a global services pointer, but it is not possible to use an envelope wrapper instance without a global services pointer. Any attempt to do so will set off an assertion.

The ability to animate parameters is a good service to provide to users, and any serious plug-in should consider using the LightWave envelope control to do so. It's easy enough to use that there's almost no reason not to include envelope support.

Brenden Mecleary is just too busy playing Asheron's Call to look into this "social life" thing everyone keeps talking about. Anyone care to explain the concept? Or just got a plug-in question, comment, or flame? E-mail bman2@ix.netcom.com.

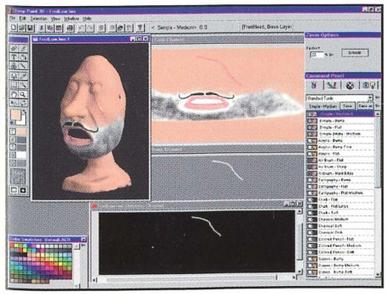

CURL * DYE * WAVE * MOUSSE * GEL * CUT
SHAVE AND A HAIRCUT ™
The finest and fastest photo-realistic hair render money can buy, can now be bought for less than a hundred bucks. More in common with ILM's Oscar winning hair render than it has with it's competitors, Shave and a Haircut is not your average "two-bit" plugin. Not so surprising when you consider Shave and a Haircut 's author wrote ILM's first hair render 6 years ago. This altogether new fractal hair rendering algorithm requires less than 32MB to render millions of hairs, and is several orders of magnitude faster than previous approaches.
But wait, there's more! Shave and Haircut comes with hair modeling tools that have no equal. In addition to automatic grooming, guide hairs are incredibly fast bi-directional IK chains. You can control them one at a time, or grab thousands and just comb away. There is collision detection built into the sculpting process as well, so hair goes around the skull - not through it.
Ok, so why is it so cheap? Because we like you.
Talk is even cheaper - try our standalone demo version for yourself.
* SHADOWING - SELF, CAST, SOFT
* MOTION BLUR
* TRANSPERENCY
* BREAKTHROUGH FRACTAL ALGORITHM
* INTERACTIVE IK MODELING
* COLLISION DETECTION
* DYNAMICS
* DEFORMS PERFECTLY WITH SKIN
* VERY LIGHT ON RAM USAGE
* SEAMLESS INTEGRATION
* NO WAITING.'' (OK, MAYBE A LITTLE...)
$99.95 (FOR NOW)
WWW.JOEALTER.COM
(Actual quickshade preview screen shots rendered in less than 15 seconds at 600x600)

Deep Paint 1.0
$795.00
Right Hemisphere (US Office)
215 West Holly, Suite H-22
Bellingham WA 98225
877-309-3204
support@righthemisphere.com
www.righthemisphere.com

Deep Paint is a standalone 3D paint program designed to work independently or as a plug-in with a variety of 3D modeling and animation software and Photoshop. Deep Paint is the successor to 4D Paint, one of the better 3D paint programs available in the past few years, so this program is more highly evolved than the 1.0 version number might lead you to believe.

If all you need to do is create a single map to fit an object in LightWave, you can probably do an adequate job with Ernie Wright's Unwrap plug-in and Photoshop. However, if you need to match up two or more maps such as color, bump, specularity, opacity, or luminosity, Deep Paint can save you a great deal of time and bother and produce much better results. This program enables you to paint any or all map types simultaneously on a 3D model, making it much easier and faster to create accurate, complex maps of up to 6000x6000 resolution.

If you have ever used Painter, Designer or another program designed for traditional painting techniques (rather than Photoshop's darkroom metaphor), Deep Paint will seem familiar. There is a broad selection of traditional tool styles including chalk, charcoal, acrylic, pencil, crayon, etc., plus canvas choices that mimic textures ranging from cloth to waterdrops-on-glass to skin. Deep Paint is also designed to work best with Wacom's Intuos or other pressure-sensitive tablets, the digital artist's input tool of choice. If the feel of the interface is important to your mapping, you should definitely choose Deep Paint over other software that does not support these input devices. After working with Deep Paint with my Wacom tablet, I won't go back to anything that requires a mouse or doesn't handle the tilt and pressure data. By comparison, painting with the mouse is like painting with a bar of soap with an oversized brush stuck through it. You simply don't have the fine control and precise sense of where each pixel is going.



The Deep Paint documentation is better than average, and certainly adequate to explain how to use the software. The printed manual is just under 200 pages, and a PDF version of the manual is included on the CD-ROM. The seven tutorials in the manual cover the features you are most likely to use, and you should be able to work through them in a few hours. The program includes context-sensitive Windows help, and there is a hotlink to a FAQ on the Right Hemisphere website.

The default installation occupies 132MB on your hard drive, but 103MB of that is filled by the Examples folder, which you can delete after working through the tutorials. Memory usage on startup is between 5 and 10MB. However, opening a project drastically increases RAM usage based on the resolution and number of maps. For example, loading the 28K-vertex Fred model with three 1608x512 maps (color, bump, and specularity) consumed an additional 40MB. Considering the unlimited levels of Undo and the very fast screen refresh, this memory hit is reasonable. The Introduction chapter of the manual is very explicit about this need for memory, including recommended minimum RAM levels for the most common map resolutions. Basically, any machine capable of running LightWave at the resolution of the maps should be capable of running Deep Paint without any problems.

Feedback while painting is very fast, and adjustable based on your preferences and your system resources. With only 128MB in my system, pushing the feedback to highest speed didn't seem to have a negative effect on anything else. Deep Paint is stable and runs well with both LightWave and Photoshop open simultaneously. My only quibble is that the default format for saving maps is TIFF, and the current version won't recall that you prefer a different format. Since I prefer TGA, I had to deal with the annoyance of changing the default TIFF to TGA every time I saved the project or exported the maps. I hope Right Hemisphere can fix this minor oversight in the next revision.

Deep Paint works in 3D mode by default. You can call up the 2D maps and paint in those if you like. However, you'll get faster response time and a more intuitive workflow if you stick with the 3D view as much as possible. Deep Paint's layers give you a great deal of flexibility to create detailed organic maps that you can edit with a minimum of fuss or wasted effort. You can use a base layer for the character's basic color, then add layers for wrinkles, skin defects, folds, scars, and whatever else makes the character interesting and unique. Combined and exported as a single image per map type, the result is easy to manage in LightWave. Preserved as layers in Deep Paint's native project format, it's easy to modify and edit. The only disadvantage is that Deep Paint's native project format is not terribly efficient. If you want to preserve all the layers and other information (which I strongly recommend!), you have to save a .DP3 project file. This is actually a file folder containing a proprietary project file plus copies of all the contributing maps. These project folders can get very large very quickly, but, if you want to be able to edit your character's maps, it's the only way to go.

Getting up to speed in Deep Paint is not a problem. Without reading the tutorials, I went straight to the one-page guide to working with LightWave and followed the instructions. I loaded a low-resolution copy of the Fred cyberscan model, and in about fifteen minutes of playing around I had the color, bump and specularity (or shininess) maps you see in Figure 1. One of the nicer features of Deep Paint is the real-time update of both bump and specular maps; getting the scar to look right took only a few strokes. Getting an acceptable skin texture was simply a matter of flood-filling the base skin color, then brushing a charcoal tool containing the top color over a veined canvas. The beard stubble was charcoal over a more literal canvas; as I said, this was the result of goofing around for a few minutes, but the results are surprisingly good for a first (and less than serious) effort. I'm looking forward to creating more detailed and complex texture map sets with Deep Paint. It's fun to work with, and avoids many of the frustrations inherent in 2D approaches to texture painting.

by Steve Pugh

By the time this issue hits the stands, Foundation Imaging will have wrapped production on *Roughnecks: The Starship Troopers Chronicles*. Barring any grave misfortunes between now and then, we will have delivered 31 of the show's 40 half-hour episodes to Sony, with Rainbow Studios, Flat Earth, and Hyperimage (a Burbank 3DS MAX facility) producing the remainder.

It was actually last year's NAB issue of *Keyframe* that featured a last-minute news item on the show, before we brought in over a hundred new animators and started waging war on those cheerful bugs from Klendathu. The campaign has most definitely been a learning experience, as I mentioned a few months ago. But as it winds down and we start to gear up for our next show, I thought a look back might be in order.

To be sure, *Roughnecks* was a great thing for Foundation. As Nietzsche said, "That which does not kill us, makes us stronger." Members of the animated-content community take us more seriously, and more shows are coming to us now that would have passed us by a year ago. The folks who waited to see if we could pull it off are starting to see us as a proven commodity.

However, some of the most significant changes have been invisible to the outside world. Animating the show meant taking an already-powerful LightWave and pushing its boundaries in ways we'd never really had to do before, and doing that on a production schedule. Our trailer for Vortex from a year prior to Roughnecks showed that we could produce an all-CG show to our standards within LightWave, but now we had to deliver almost fifteen hours of top-notch content. Clearly, some nuts and bolts would need tightening.

Before *Roughnecks*, our render controller was effective and efficient, but over the past year FIRE (Foundation Imaging's Render Environment - and a quick thanks again to Ron and Paul for choosing such an acronym-friendly name for the company!) has really grown up. The transition to multiple animation teams led to features like job and engine grouping, animator- and supervisor-permissions, and batch-emailing of completion notices.

The show's "multiple houses" approach required a separate Content Directory, which we had never used before. This required the ability to switch Content folders and restart Screamernet remotely on over 300 nodes. Little features crept in, like the ability to browse a rendering scene's frames or email its animator with a single click.

We also found ourselves seeking outside help for some of *Roughnecks*' bigger challenges. Some of the LightWave development community's brightest talents were enlisted to help defeat the bugs we found in custom volumetric tracer fire, CG asset-tracking software, and lip-sync utilities; and to present an extended workshop on clever ways to use a particular expression package. Tools and features we never considered before have become second nature.

Now that our next show is coming up the driveway, we're looking at LightWave 6.0. Its power may well render some of the past year's innovations obsolete. To be sure, it will eliminate many of the headaches we were unable to find a crafty way around. However, every show is different. I look forward to the chance to sit here a year from now and to be amazed by what things yet to be imagined have become indispensable.

Steve Pugh is the General Manager at Foundation Imaging. If he's not daydreaming or fighting a mental block, you can contact him through their website, www.foundation-i.com.

• Uncompressed D1
• SpeedRazor™ SE Non-linear Editor
• Aura™ Video Paint
• LightWave® VT

www.videotoaster.com
Call (800) Toaster
Outside the U.S. call (210) 370-8000

NewTek

Completely redesigned and integrated into LightWave
Advanced Collision Detection and Response
New Constraints - Spring, Rod, Joints, Rope, Motors, and More
Varying Force Generators

Coming soon from the plug-in experts!